

全集成自动化(TIA) 解决方案培训教材

第一部分第四章

CPU 315-2DP 的编程

这个手册由西门子自动化与驱动集团教育合作部(automation and drive technology, Siemens A&D Cooperates with Education)以培训为目的编写。西门子对其内容不做任何形式的保证。

手册的传播或者复制，包括其内容的使用与发表，仅作为公共教育及职业培训之用。

其他情况需要西门子自动化与驱动集团教育合作部的书面许可（Knust 先生，E-Mail:michael.knust@hvr.siemens.de）。违者必究。西门子保留所有权力，包括翻译，以及专利权、实用新型或外观设计专有权。

感谢 Michael Dziallas Engineering 公司、职业学校的教师们，和其他有关朋友为本手册的编写做出的贡献。

目录:

1. 前言.....	4
2. CPU 315-2DP使用的注意事项.....	6
3. 如何生成CPU 315-2DP的硬件组态.....	7
4. STEP7 程序的编写.....	19
5. STEP-7 程序的调试.....	22

下列符号代表的含义:



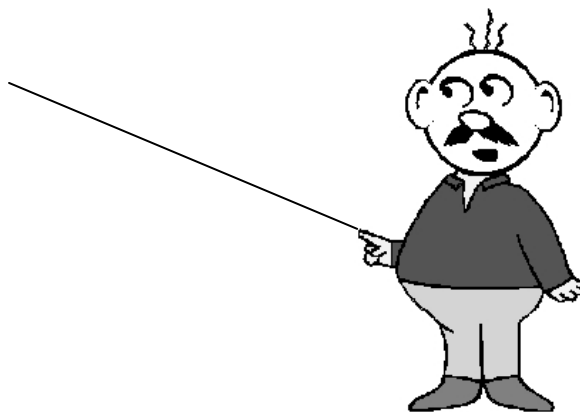
信息



举例练习

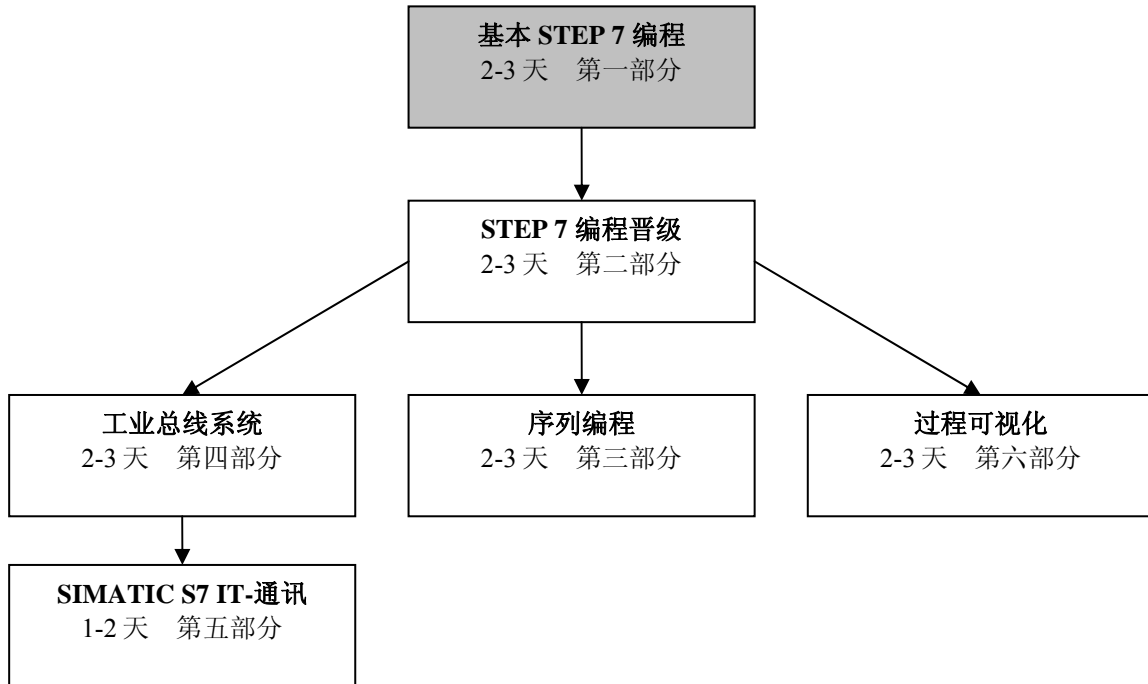


注意



1. 前言

第四部分的内容是配合 STEP 7 的基本编程课程设置的。图示如下：



学习目标：

在这一章中，读者应该了解CPU 315-2DP的硬件组态如何生成，STEP 7程序的编写和调试。课程包含了一些基本的步骤，并通过详实的例子来说明这些基本的编程原则。

- STEP 7项目的应用
- CPU 315-2DP 硬件组态的生成
- 编写一个STEP 7程序
- STEP 7程序的调试

基本条件：

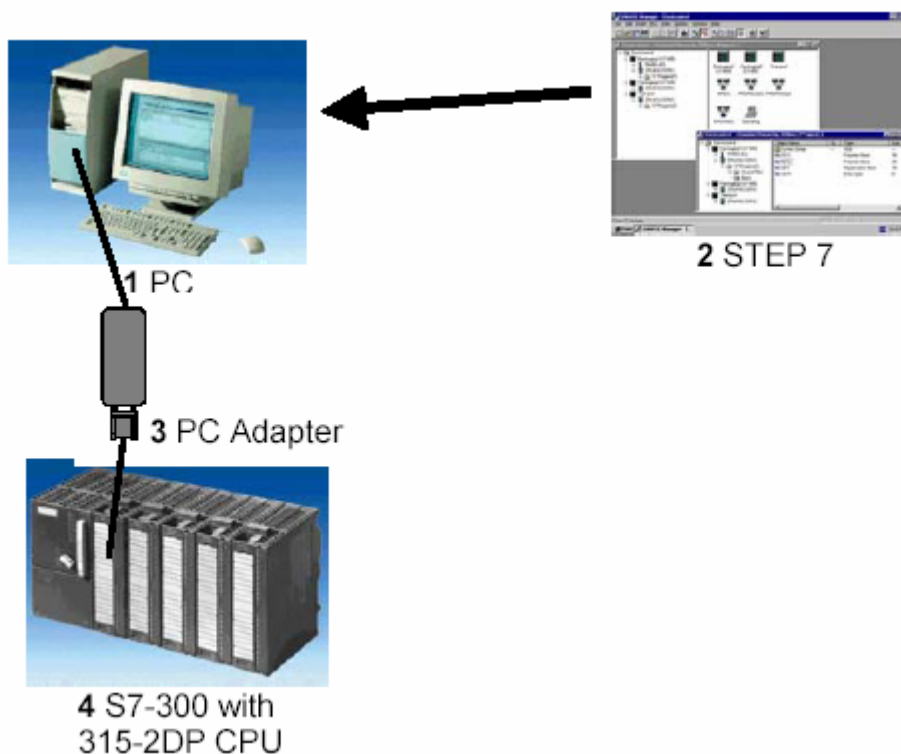
为了这部分内容的顺利进行，我们希望读者具备以下的基础知识：

- Windows 95/98/2000/ME/NET的基本操作知识
- 使用STEP 7进行PLC编程的一些基本知识

需要的硬件和软件：

- 1 PC , Windows 95/98/2000/ME/NET的操作系统，以及
最小：133MHz，64MB 的RAM ， 65MB的空余磁盘空间。
最佳： 500MHz， 128MB 的 RAM， 65MB 的空余磁盘空间。
- 2 STEP 7 5. x 软件。
- 3 一个PC用MPI接口。
- 4 一个装有CPU 315-2DP的 PLC SIMATIC S7—300，
组态举例：

- 电源： PS 307 2A
- CPU : CPU 315-2DP
- 数字式输入： DI 16x DC 24V
- 数字式输出： DO 16x DC 24V/0.5A



2. CPU 315-2DP使用的注意事项



CPU 315-2DP 是集成了 PROFIBUS-DP 接口的 CPU。CPU 315-2DP 可以使用以下 PROFIBUS 协议文件。

- 遵守 EN 50170 的主站 DP 接口。
- 遵守 EN 50170 的从站 DP 接口。

PROFIBUS-DP 是一个用于和辅助外围设备或现场设备连接的协议，这个协议的响应时间很短。CPU 还可以实现更进一步的功能，即将输入输出模块的地址参数化，这样就可以通过改变参数来设置输入输出模块的地址。

符合下列技术参数的项目程序，其执行效率将足以满足培训之用。

- 16k **statements**. 48k 的工作空间 80k 的 **build space**
- 1024 Byte DI/DO
- 128 Byte AI/AO
- 每执行 1000 条指令需要 0.3 .ms
- 64 个计数器
- 128 个定时器
- 2048 位 存储器

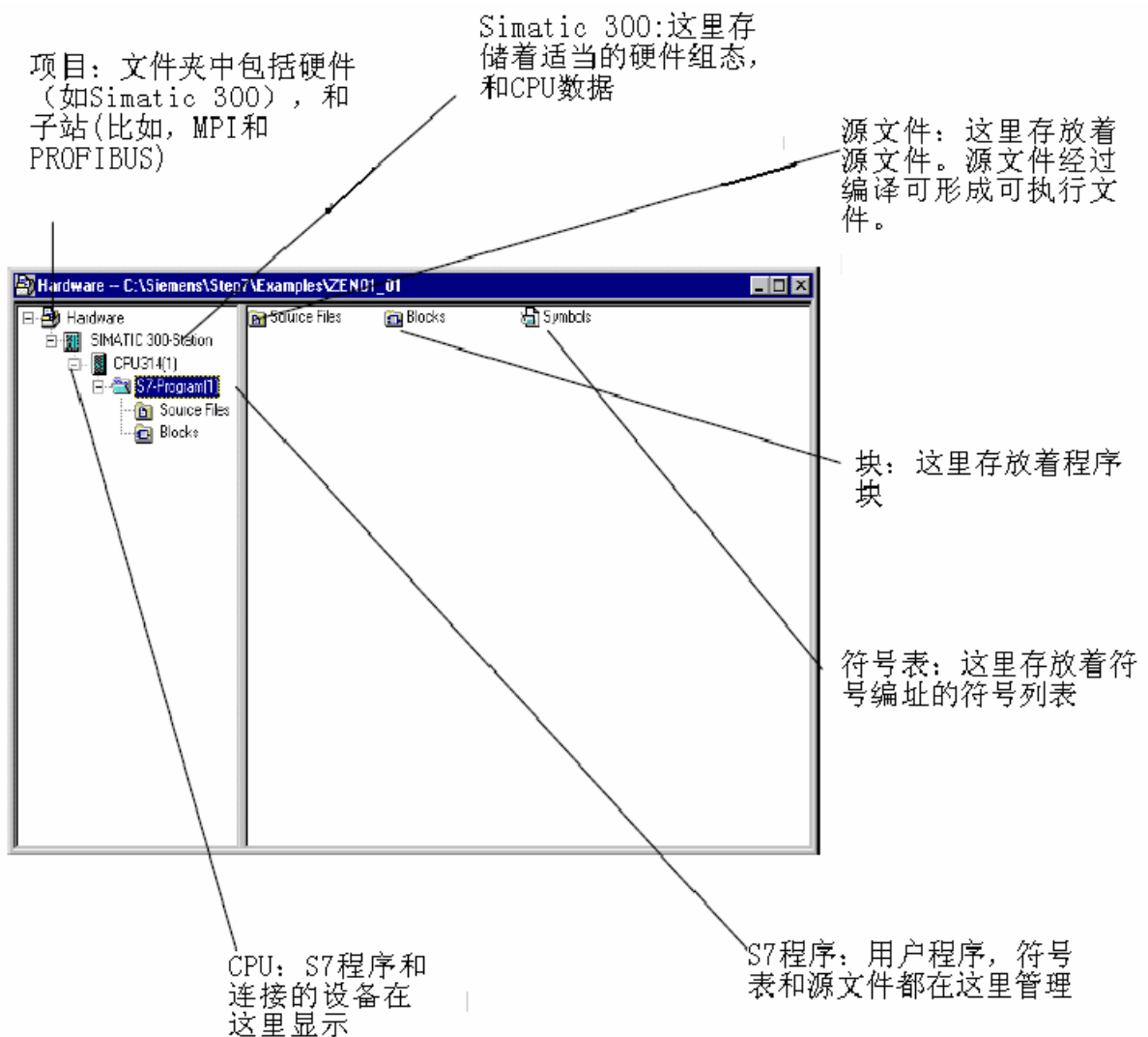
3. 如何生成CPU 315-2DP的硬件组态



在 STEP7 中我们用 SIMATIC Manager 来进行文件管理。这里的样例程序段只需要双击就可以被拷贝或是调用，以使用其它工具完成进一步的编程工作。操作符合常见的 Windows 95/98/2000/ME/NET 的操作标准。

在文件夹 SIMATIC 300 station and CPU 中，说明了 PLC 的硬件结构。因此，我们可以具体了解到每一个项目所包含的硬件设备。

在 STEP7 中，每一个项目都有一个确定的结构。程序被储存在以下的目录中：



这里举了一个 CPU 315-2DP 组态的例子，还应该再设置时钟存储器，调整输入输出模块的地址。



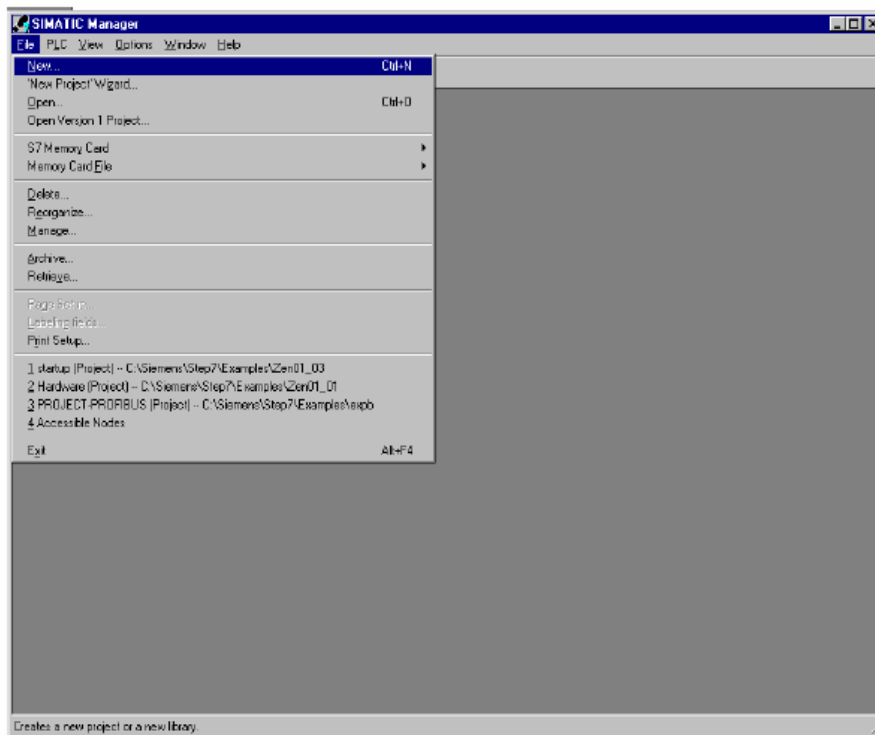
用户可以按照下列步骤操作，来生成一个项目文件，并编写程序。

1. STEP7 中的主要工具是 SIMATIC Manager，可以通过双击以下图标来打开。
(→SIMATIC Manager)



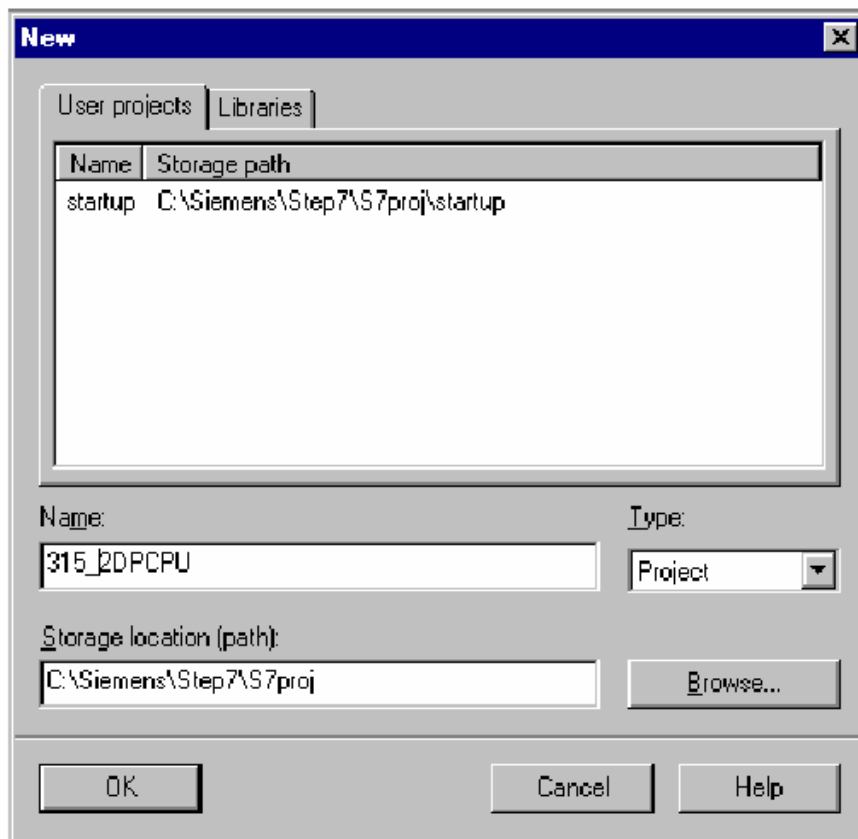
SIMATIC Manager

2. STEP7 的程序是在项目文件中管理的。每一个项目文件通过点击File菜单中的 New选项来产生。(→ File→ New)

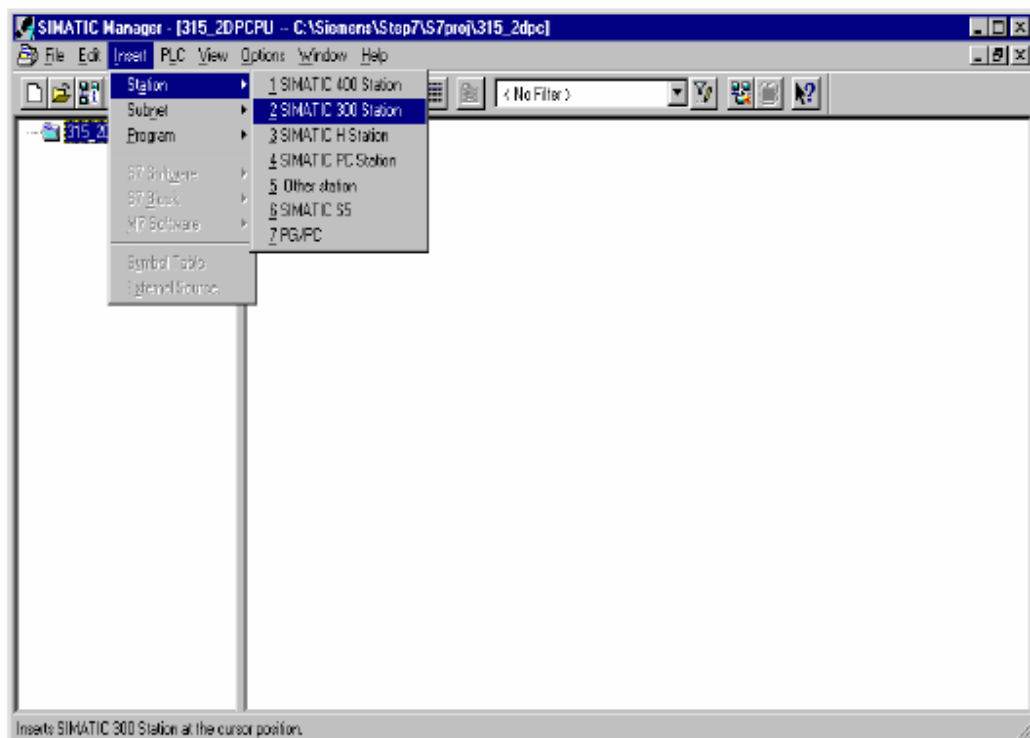




3. 将项目命名为 315_2DPCPU。(→3152_DPCPU→ OK)

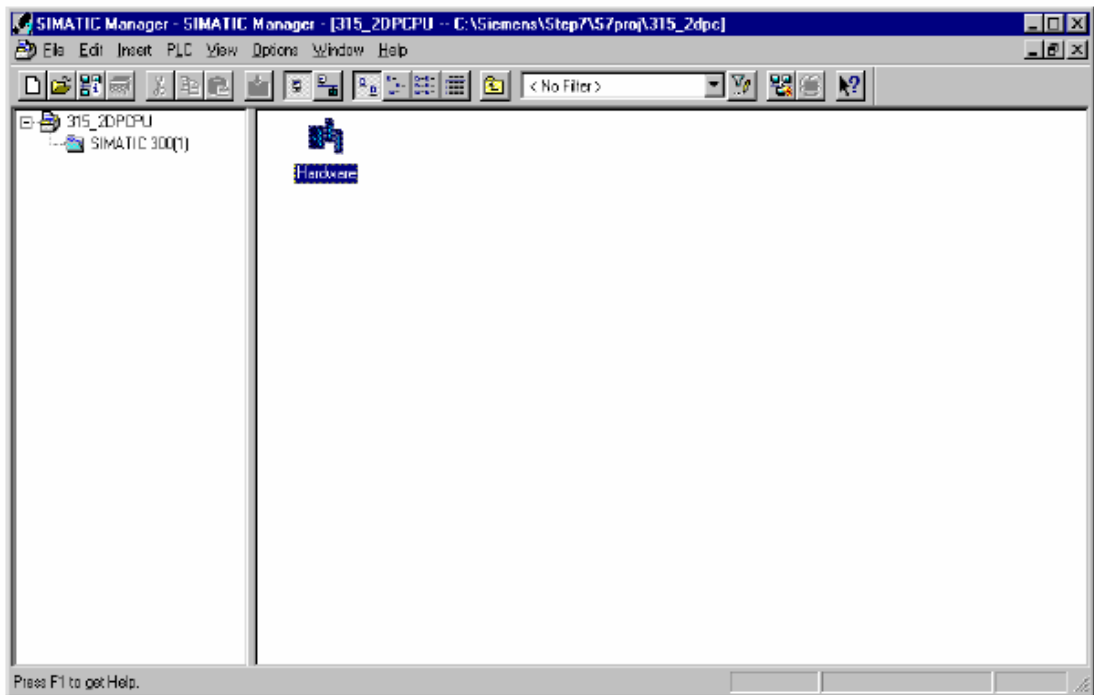


4. 载入站点 SIMATIC 300-Station。(→ Insert → Station → SIMATIC 300-Station)



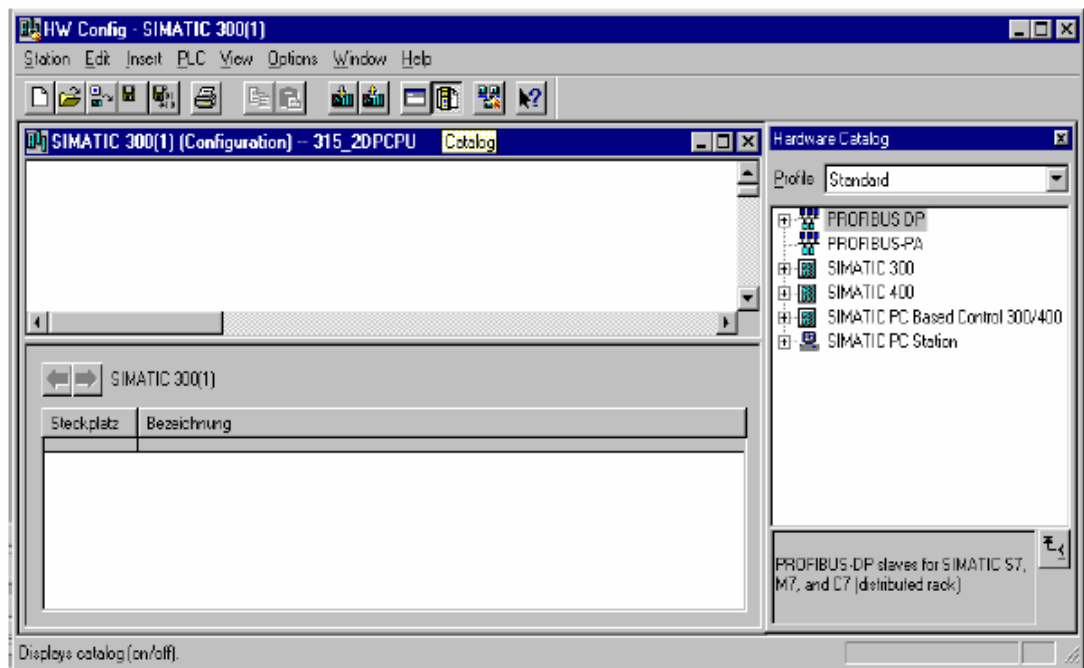


5. 双击 Hardware 图标，打开组态工具箱。



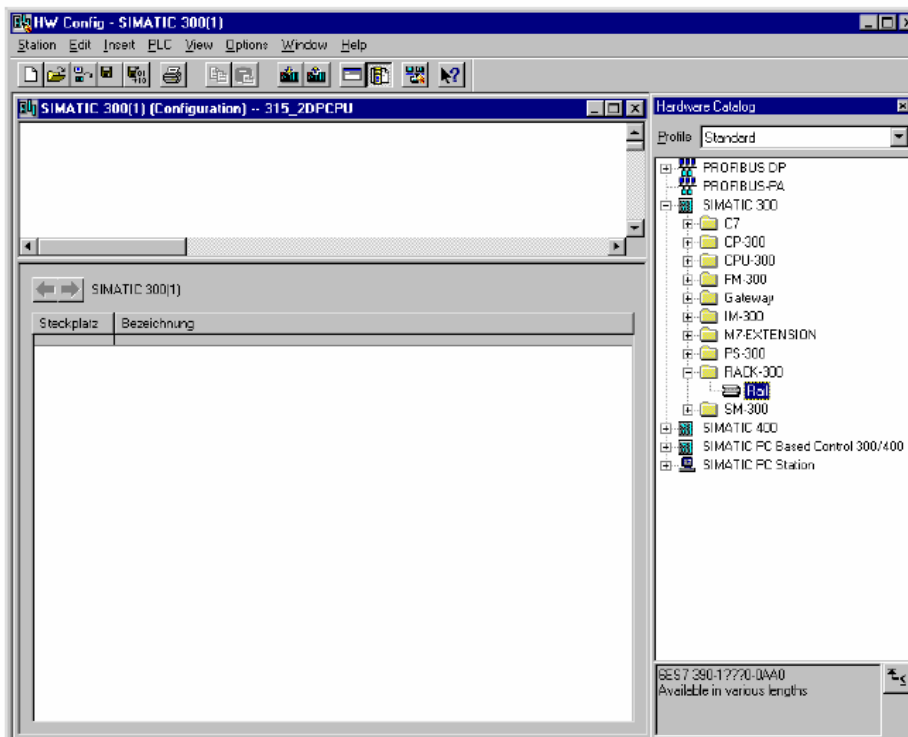
6. 双击图标 ，打开硬件列表。（→ ）

硬件组态内容分成以下的模块-PROFIBUS-DP,SIMATIC 300,SIMATIC 400 和 SIMATIC PC Based Control。搭建一个项目所需的所有模块，数据块和接口都显示在这里。





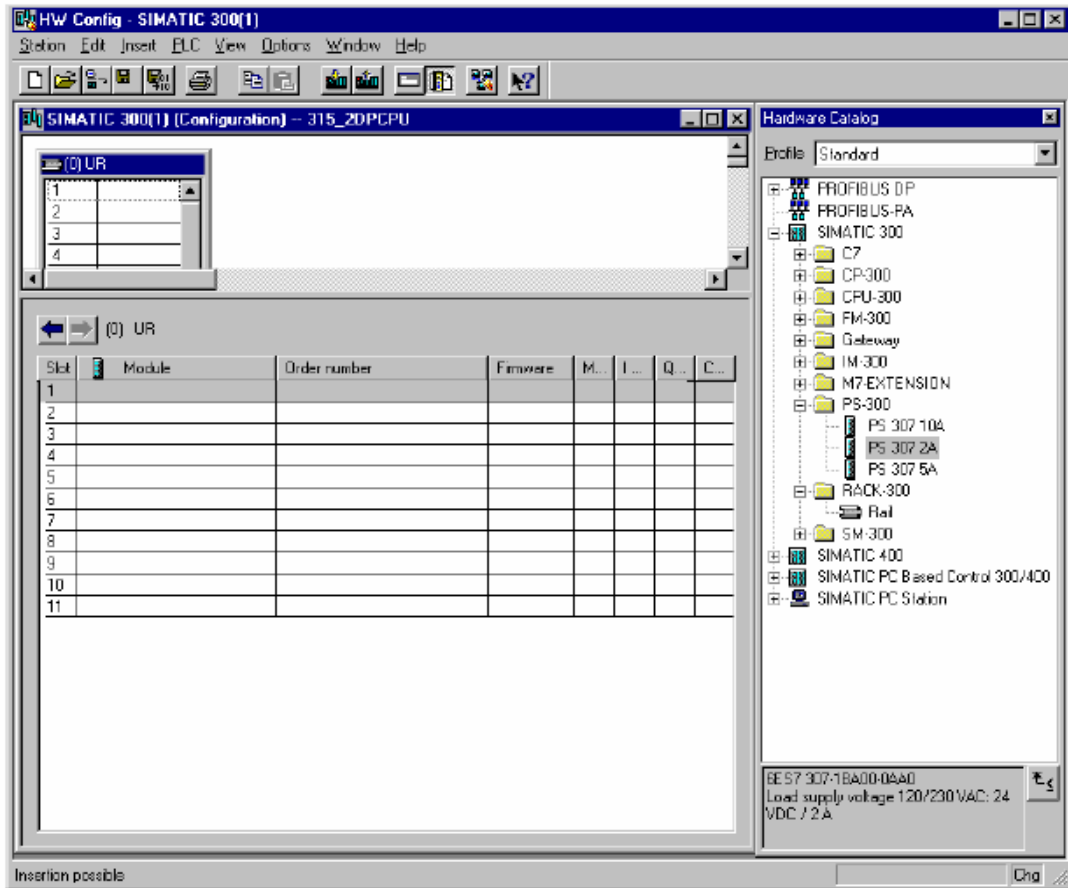
7. 双击SIMATIC菜单下的Rail。（ → SIMATIC 300 → RACK-300 → Rail ）



之后，RACK0结构的组态模板就自动生成了。



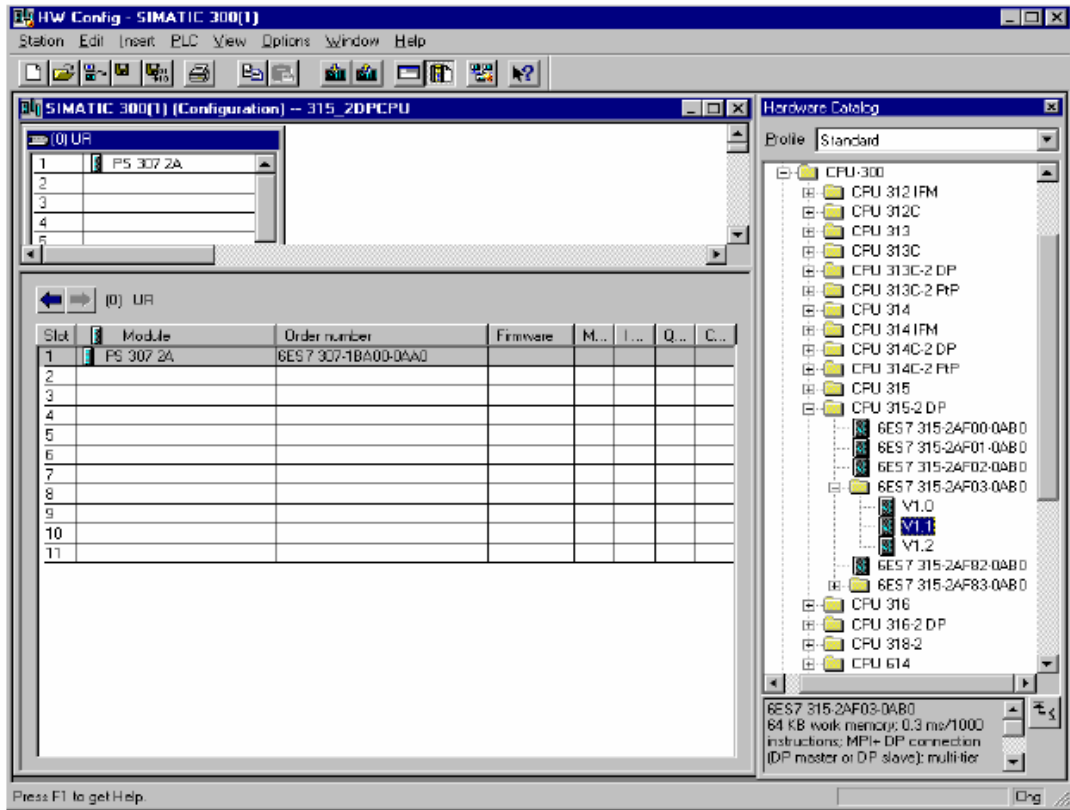
8. 现在，样例 rack 中的所有模块都可以从硬件列表选中，并插入组态模板。用户必须点击选中想要添加模块的图标，按住鼠标，拖动到组态模板中。我们从添加电源模块 PS 307 2A 开始。



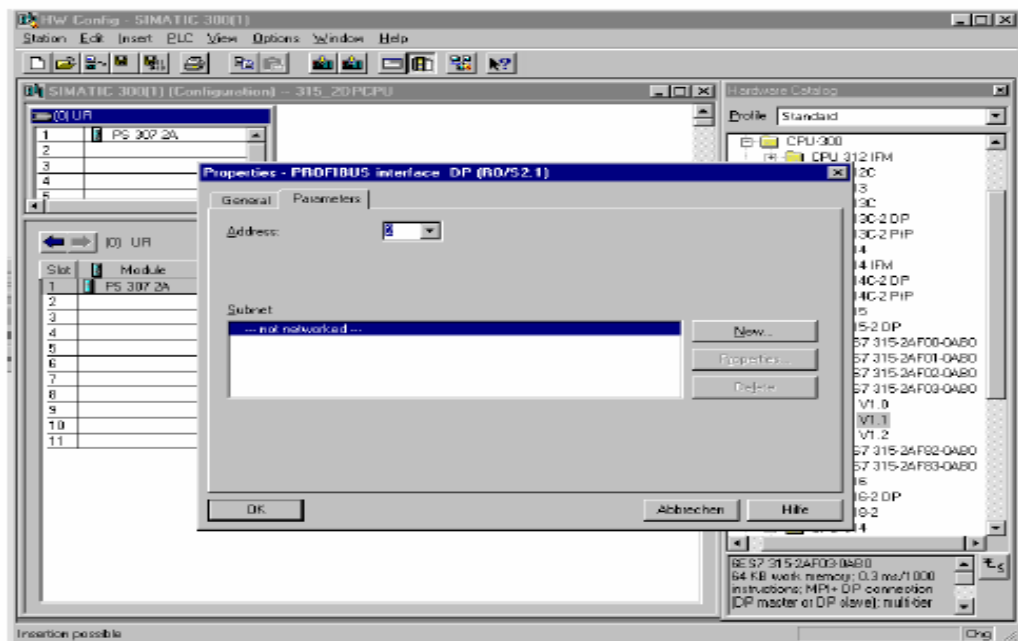
注意：按照上述步骤操作时，如果你的硬件和上面显示的不同，你也必须从列表中选中适当的模块，将他们插入到导轨中。每个模块的订货号，显示在列表的页脚方框内。



9. 下一步，我们拖动CPU 315-2DP到列表中第二位置，这样就可以读取CPU的订货号和版本号了。（→ SIMATIC 300 → CPU-300 → CPU 315-2DP → 6ES7 315-2AF03-0AB0 → V1.1）

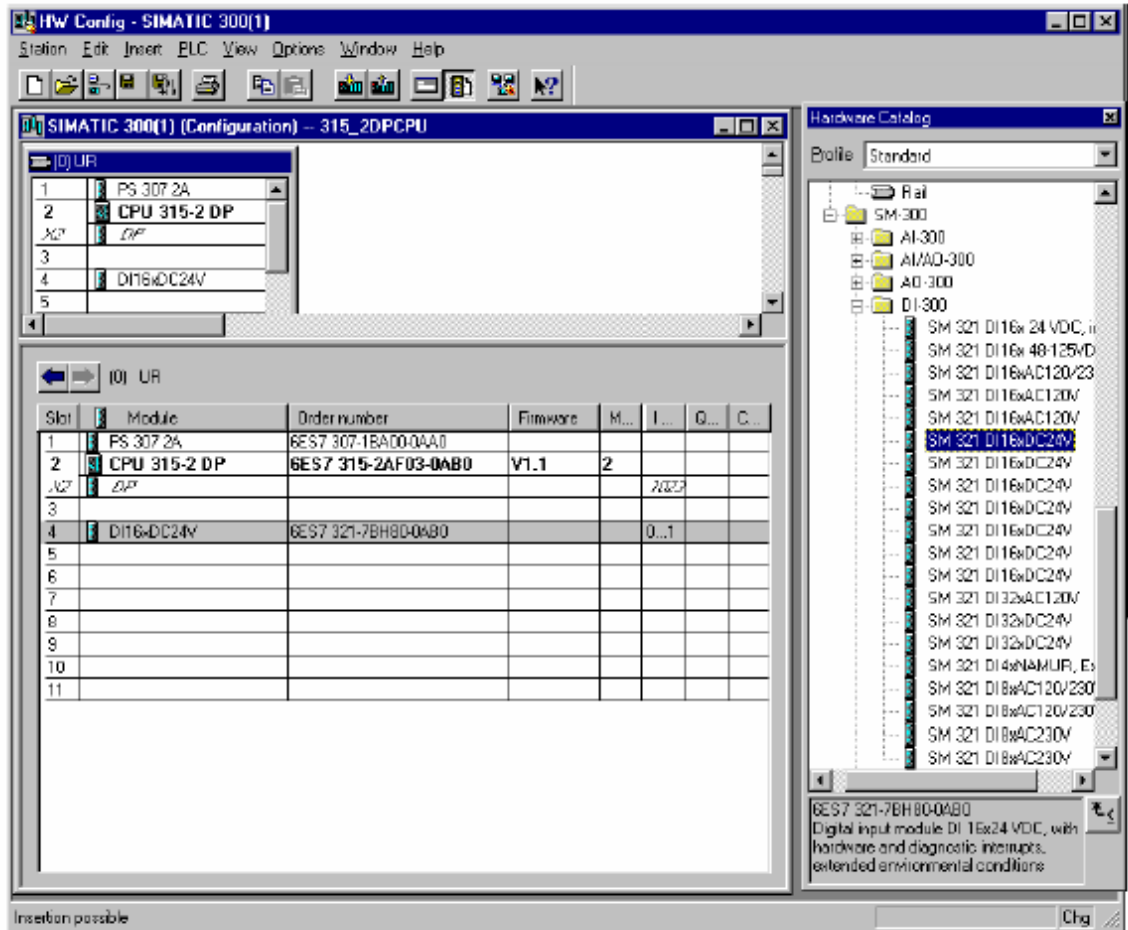


10. 可以在下面的对话框中调整集成PROFIBUS接口，由于我们在这里不对其进行调整，所以直接点击 OK。（→ OK）





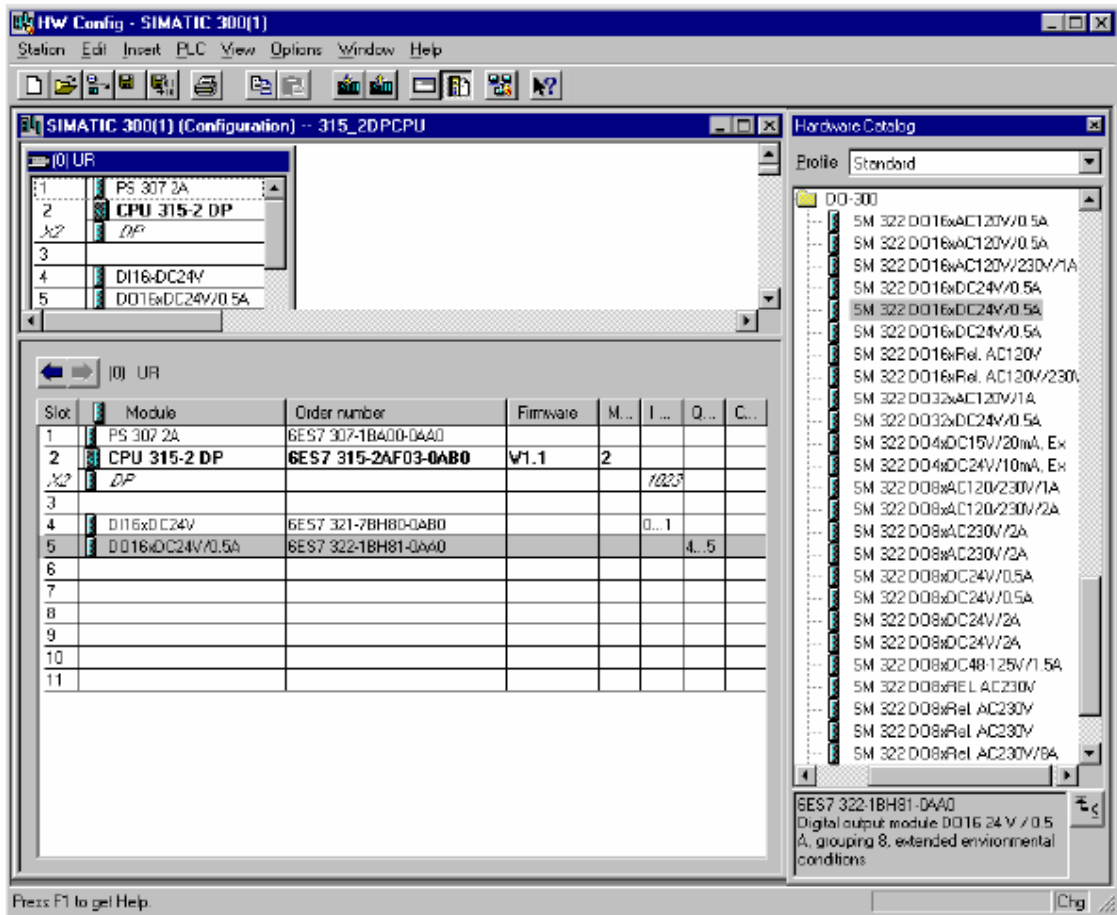
11. 下一步，我们拖动16输入的输入模块到列表中第四个位置。模块的订货号显示在第一格中。（→ SIMATIC 300 → SM300 → DI-300 → SM 321DI16xDC24V）



注意：列表中的第三个位置一般都为连接模块预留。模块的订货号显示在右下角的小窗口中。



12. 下一步，我们拖动16输出的输出模块到列表中的第五位置。模块的订货号显示在第一格中。（→ SIMATIC 300 → SM300 → DO-300 → SM 322 DO16xDC24V/0.5A）



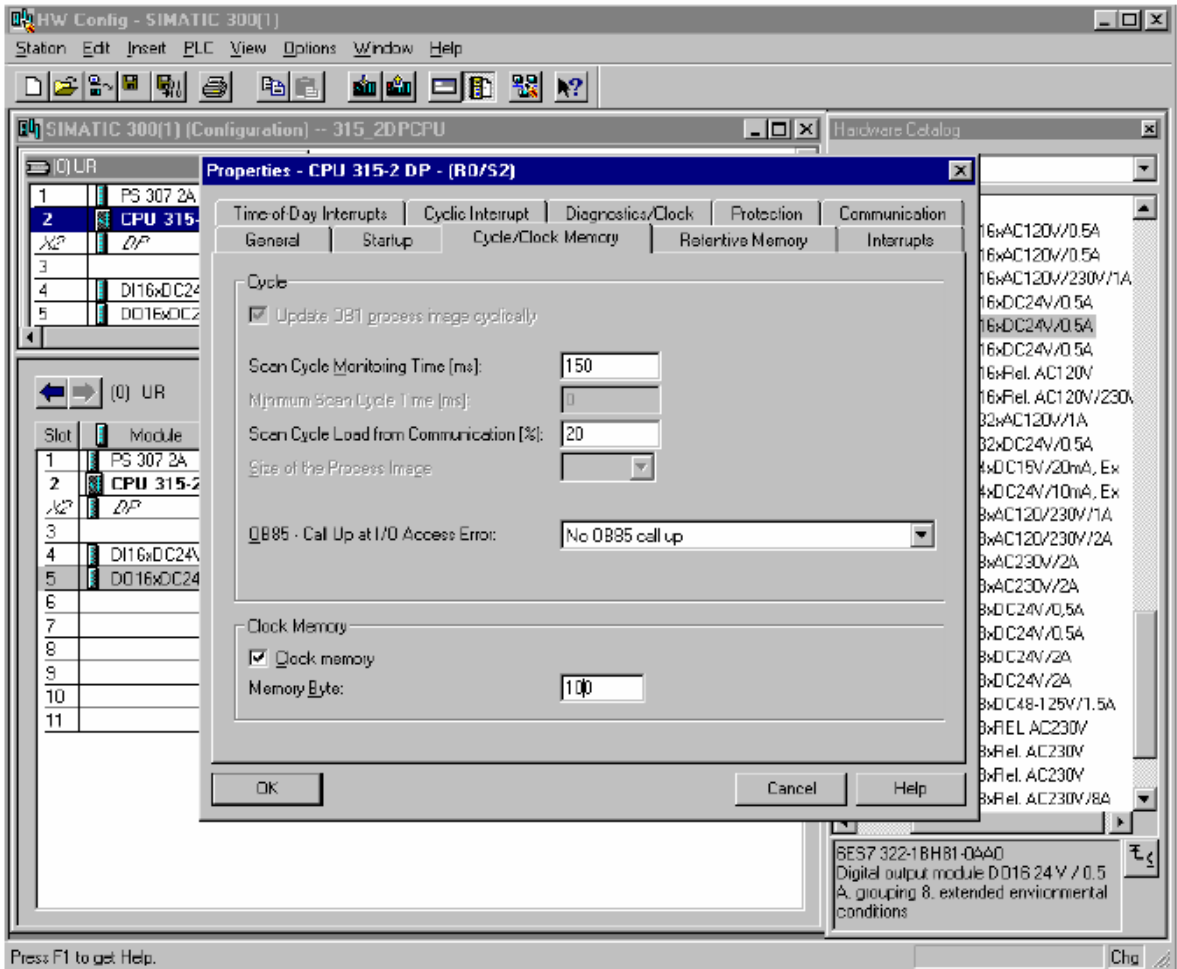
注意：模块的订货号显示在右下角的小窗口中。



13. 一些模块的属性可以在这里改变。

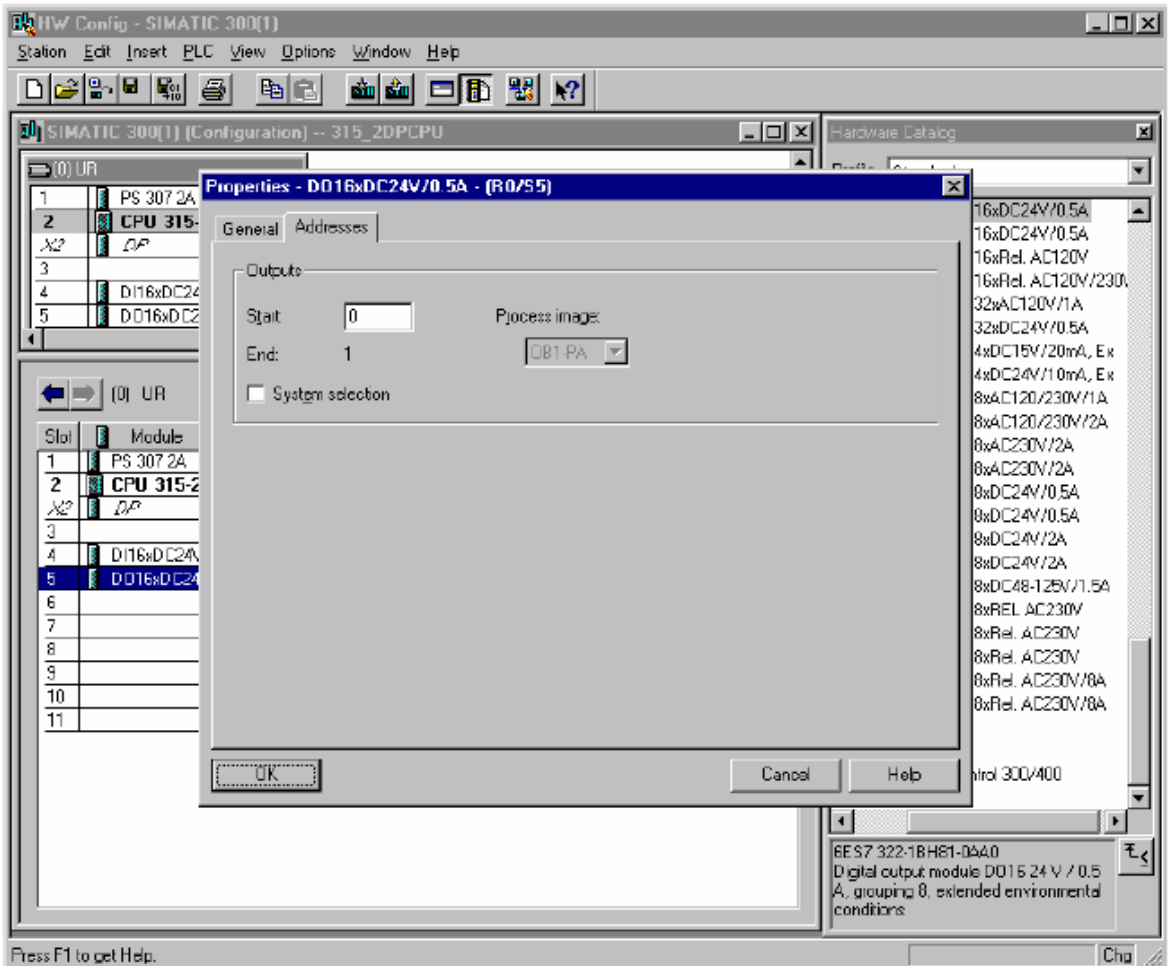
(→ Right click CPU 315-2DP module → insert_object properties → OK)

所有的 CPU 都可以设定时钟存储器。比如，将时钟存储器的内存地址设为 MB100 (→ Cycle/Clock memory → Clock memory → Memory byte 100)







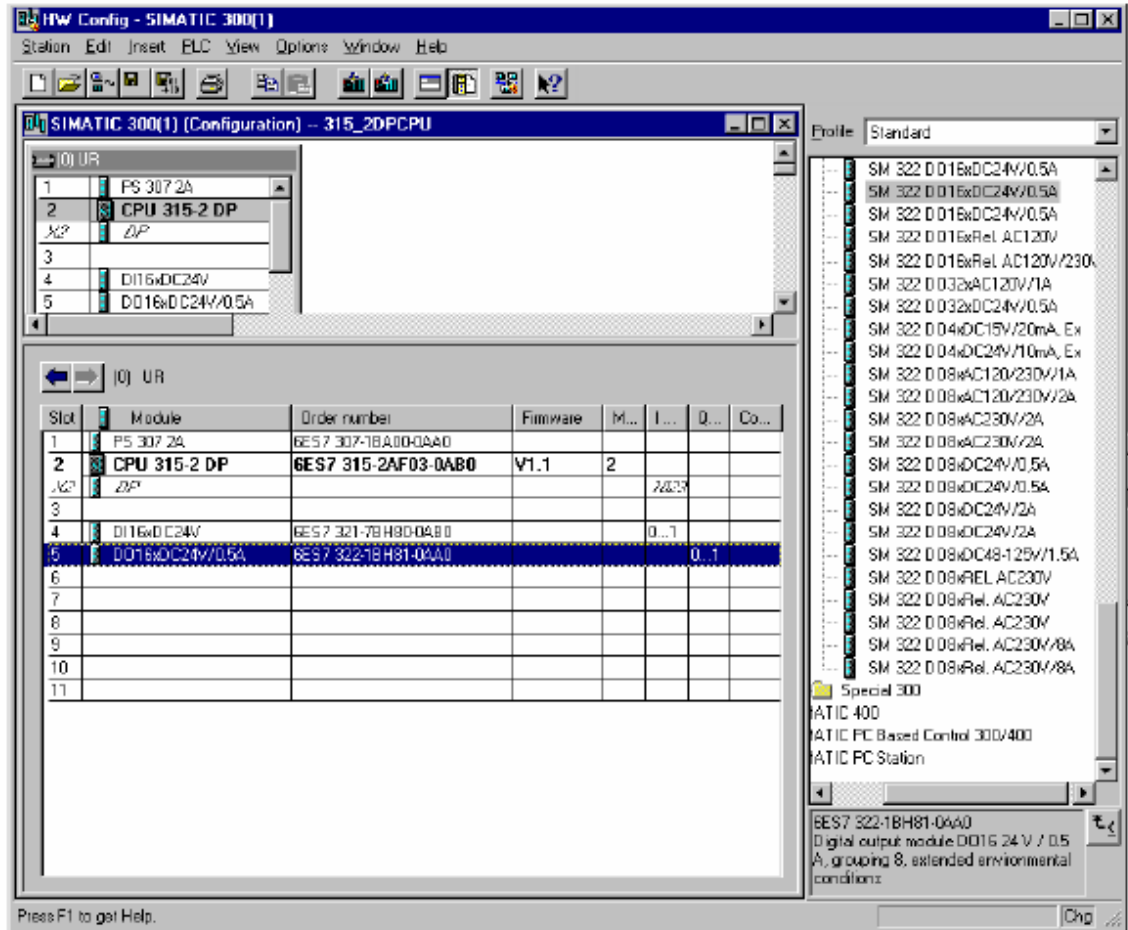
14. 输入输出模块的地址只有通过S7-300CPU的集成PROFIBUS接口才能改变。双击相应的模块图标，可以在‘Addresses’寄存器中调整输入输出模块的地址。在每一个项目中都应当注意这些地址（否则自动设置的默认地址将影响列表中的选项组合）。（→ DO 16xDC24V/0.5A →Addresses →uncheck System selection → 0 → OK）





15. 通过点击  和  图标，硬件组态模板可以保存，传送，下载到PLC中。

且传送过程中，CPU的开关必须置于STOP状态！（→  → ）



4. STEP7 程序的编写



例子中的程序使用语句表（STL）编写的，只有两句，可以用调试工具调试。在这个程序中，存在 MB100 中的时钟存储器频率值将作为数据输出。

语句表： MB100 clock clock memory byte
 QB 0 QB ouput display

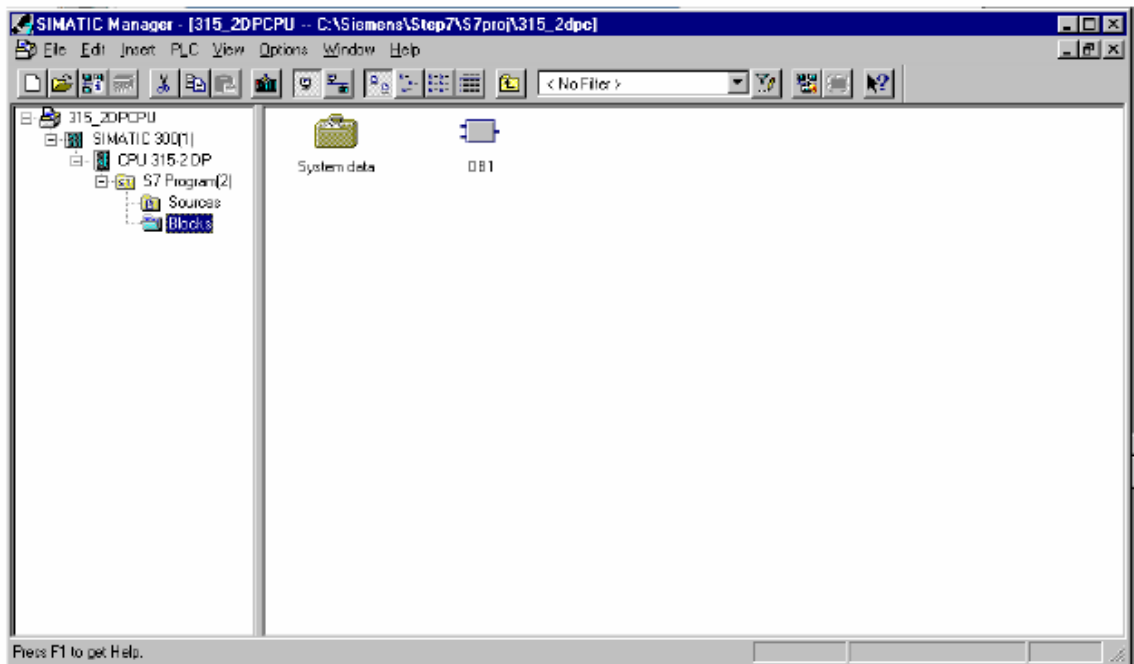


时钟存储器中的每一位都需设定一个周期/频率值。以下的设定就是适用的：

位	7	6	5	4	3	2	1	0
周期	2	1. 6	1	0. 8	0. 5	0. 4	0. 2	0. 1
频率	0. 5	0. 625	1	1. 25	2	2. 5	5	10

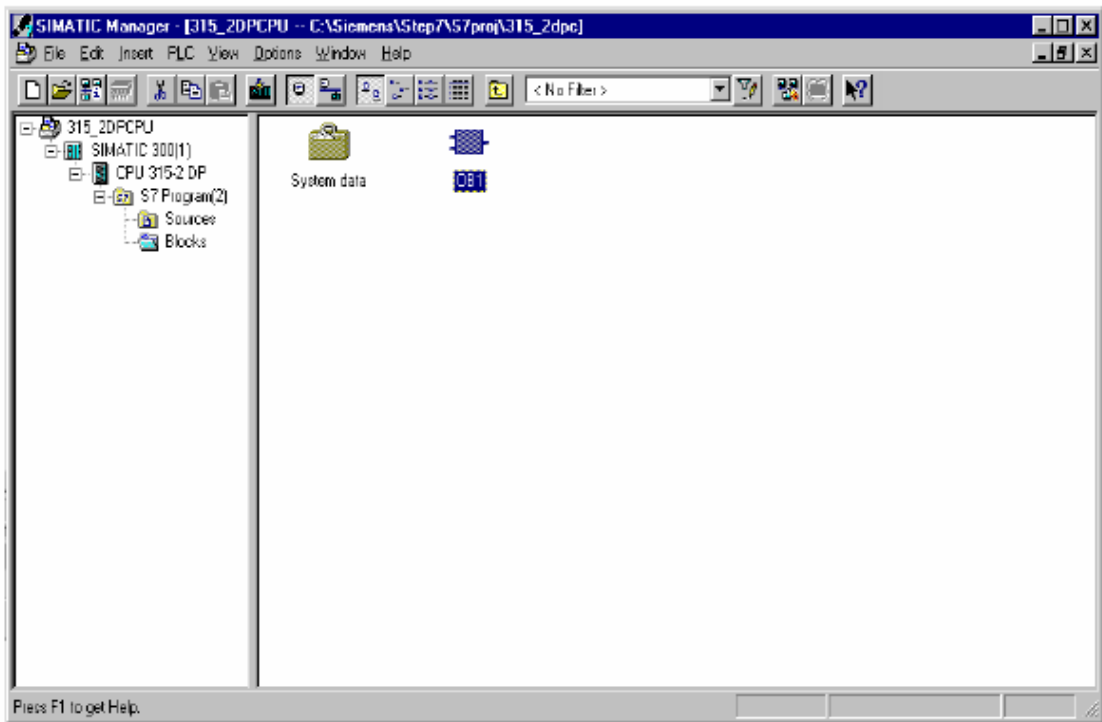


16. 在SIMATIC Manager中选中块文件夹。（→SIMATIC Manager →Blocks）

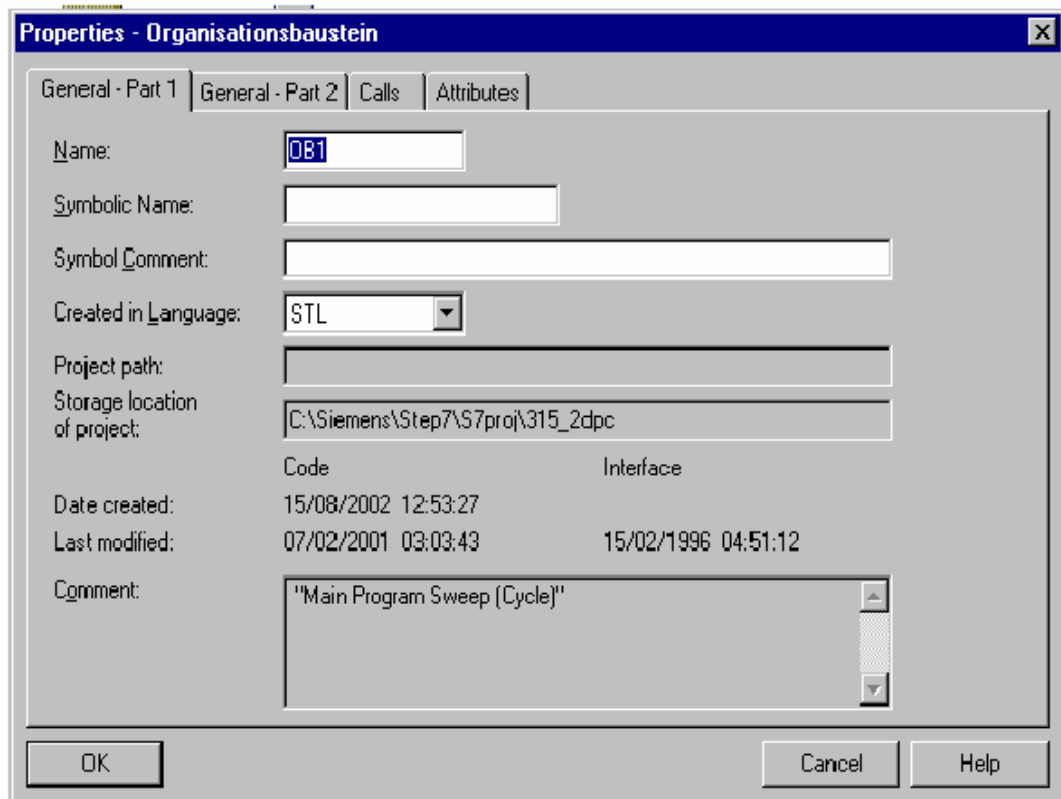




17. 在SIMATIC Manager中双击 Blocks区的 OB1。(→OB1)



18. 点击 OK，接受OB1的选项设置。(→OK)



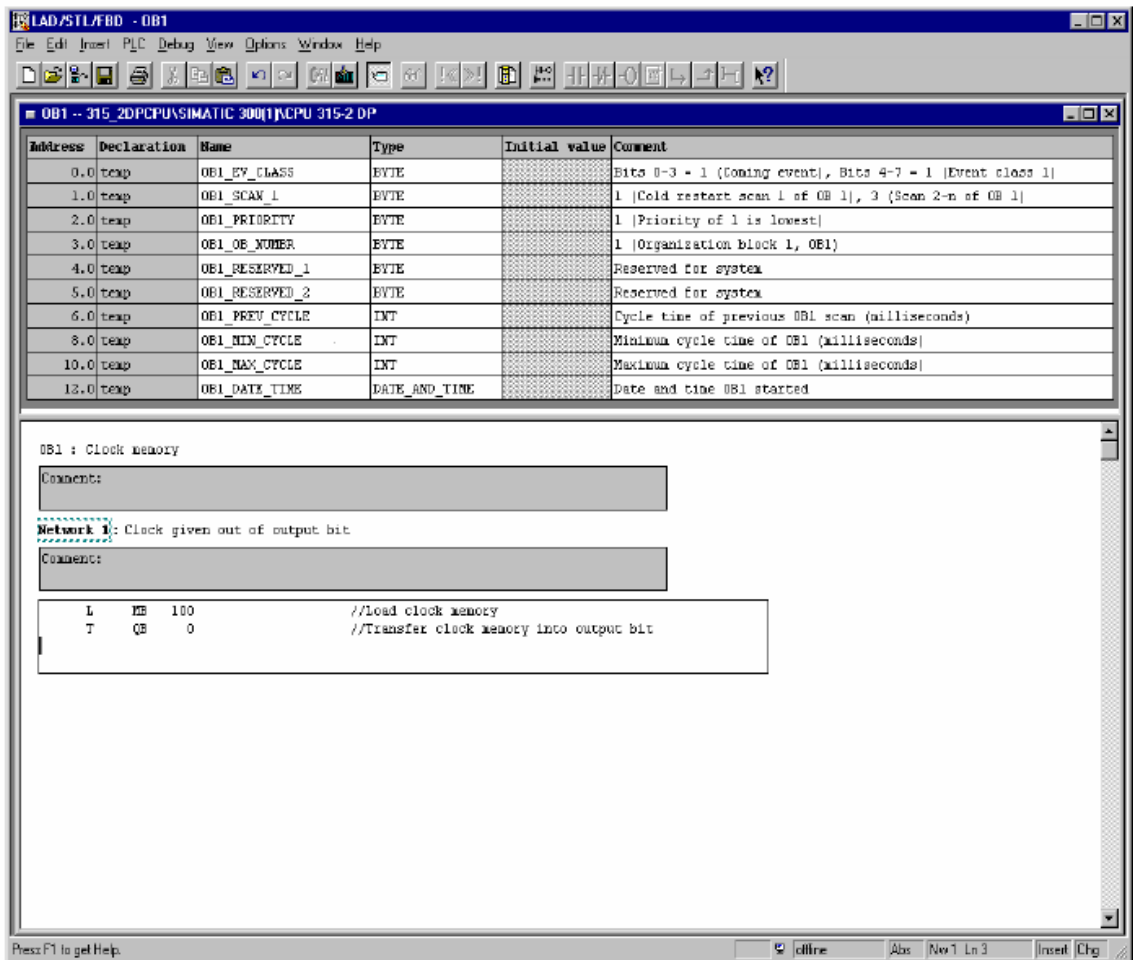


1. 编辑器配有 **LAD** (梯形图), **STL** (语句表), **FBD** (功能块) 三种编辑方式。在程序块中按照相应的方式编辑程序。激活第一个 **Network**, 并打开块**OB1**。然后开始写STEP 7 程序。每一个独立的STEP7 程序都可以分成若干

Network。可以通过点击  图标来产生新的**Network**。



注意: 程序文档的内容和程序注释通过分隔符 ‘//’ 来分开。



在 **Network** :

```

L   MB 100      //Line 1
T   QB 0        //Line 2
    
```

Line1激活了时钟存储器, **Line2**将相应的频率数据传送到输出位上。这时, **8**位的输出将依据时钟存储器的不同频率而闪烁。



注意: 输出的地址将因硬件组态的不同而不同。

5. STEP-7 程序的调试



调试后的程序才能被PLC读取。作为例子，这里我们只调试 OB1。



19. 点击  将组织块存盘，并点击  将程序下载至PLC，下载时，CPU开关必须置于STOP状态。

The screenshot shows the 'OB1 - 315-2DPCPU SIMATIC 300 IN CPU 315-2 DP' configuration window. It contains a table of parameters and a ladder logic network below it.

Address	Declaration	Name	Type	Initial value	Comment
0.0	temp	OB1_EV_CLASS	BYTE		Bits 0-3 = 1 (Coming event), Bits 4-7 = 1 (Event class 1)
1.0	temp	OB1_SCAN_1	BYTE		1 (Cold restart scan 1 of OB 1), 3 (Scan 2-n of OB 1)
2.0	temp	OB1_PRIORITY	BYTE		1 (Priority of 1 is lowest)
3.0	temp	OB1_OB_NUMBER	BYTE		1 (Organization block 1, OB1)
4.0	temp	OB1_RESERVED_1	BYTE		Reserved for system
5.0	temp	OB1_RESERVED_2	BYTE		Reserved for system
6.0	temp	OB1_PREV_CYCLE	INT		Cycle time of previous OB1 scan (milliseconds)
8.0	temp	OB1_MIN_CYCLE	INT		Minimum cycle time of OB1 (milliseconds)
10.0	temp	OB1_MAX_CYCLE	INT		Maximum cycle time of OB1 (milliseconds)
12.0	temp	OB1_DATE_TIME	DATE AND TIME		Date and time OB1 started

OB1 : Clock memory

Comment:


Network 1: Clock given out of output bit

Comment:

```

L   MB 100           //Load clock memory
T   OB  0            //Transfer clock memory into output bit
    
```



20. 将CPU开关置于‘RUN’状态，程序将被执行。执行之后，通过点击  可

以查看程序执行的状况。（→ ）

The screenshot shows the SIMATIC Manager interface for OB1. The top part is a table with the following data:

Address	Declaration	Name	Type	Initial value	Comment
0.0	temp	OB1_EV_CLASS	BYTE		Bits 0-3 = 1 (Coming event), Bits 4-7 = 1 (Event class 1)
1.0	temp	OB1_SCAN_1	BYTE		1 (Cold restart scan 1 of OB 1), 3 (Scan 2-n of OB 1)
2.0	temp	OB1_PRIORITY	BYTE		1 (Priority of 1 is lowest)
3.0	temp	OB1_OR_NUMBER	BYTE		1 (Organization block 1, OB1)
4.0	temp	OB1_RESERVED_1	BYTE		Reserved for system
5.0	temp	OB1_RESERVED_2	BYTE		Reserved for system
6.0	temp	OB1_PREV_CYCLE	INT		Cycle time of previous OB1 scan (milliseconds)
8.0	temp	OB1_MIN_CYCLE	INT		Minimum cycle time of OB1 (milliseconds)
10.0	temp	OB1_MAX_CYCLE	INT		Maximum cycle time of OB1 (milliseconds)
12.0	temp	OB1_DATE_TIME	DATE AND TIME		Date and time OB1 started

Below the table, the ladder logic network is shown:

```

OB1 : Clock memory
Comment:
Network 1: Clock given out of output bit
Comment:
L MD 100 //Load clock memory
T QB 0 //Transfer clock memory into output bit
    
```