EPICS using SLS s7plc driver

PLC configurations

National Synchrotron Radiation Research Center

Radio Frequency Group

Yu-Han Lin

In this documents, the following items are used:

Software-

1) EPICS Base 3.14.8.2 (http://www.aps.anl.gov/epics/base/R3-14/index.php)
2) Step 7 – SIMATIC Manager V5.2
3) SLS s7 driver (http://epics.web.psi.ch/software/s7plc/)

Hardware-

1) Siemens S7-300 PLC
2) Siemens CP343 Communication module

Configuration of Siemens S7 PLC

Step 1: Create an empty project.

Execute the "SIMATIC Manager" and create a new project called "EPICS_PLC" as shown in Figure 1.

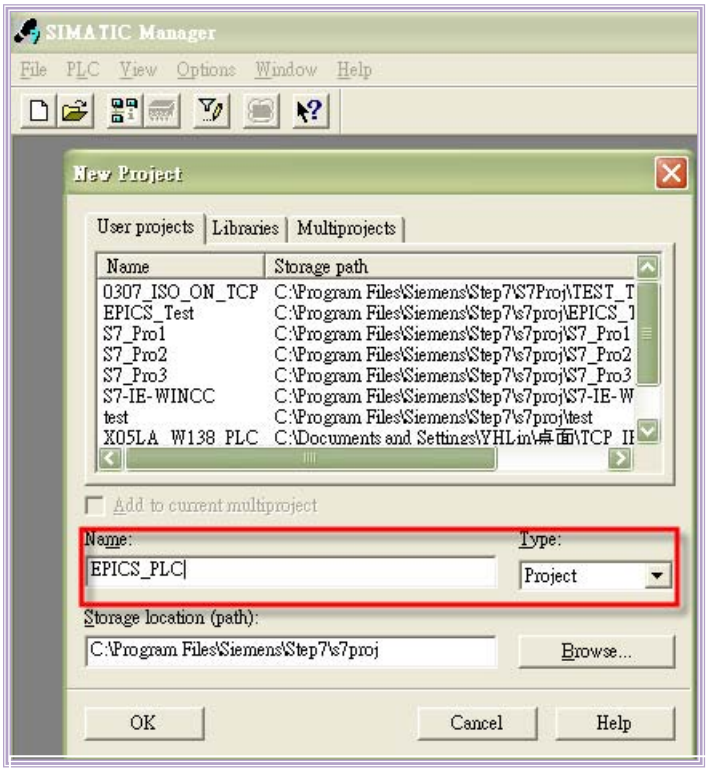


Figure 1

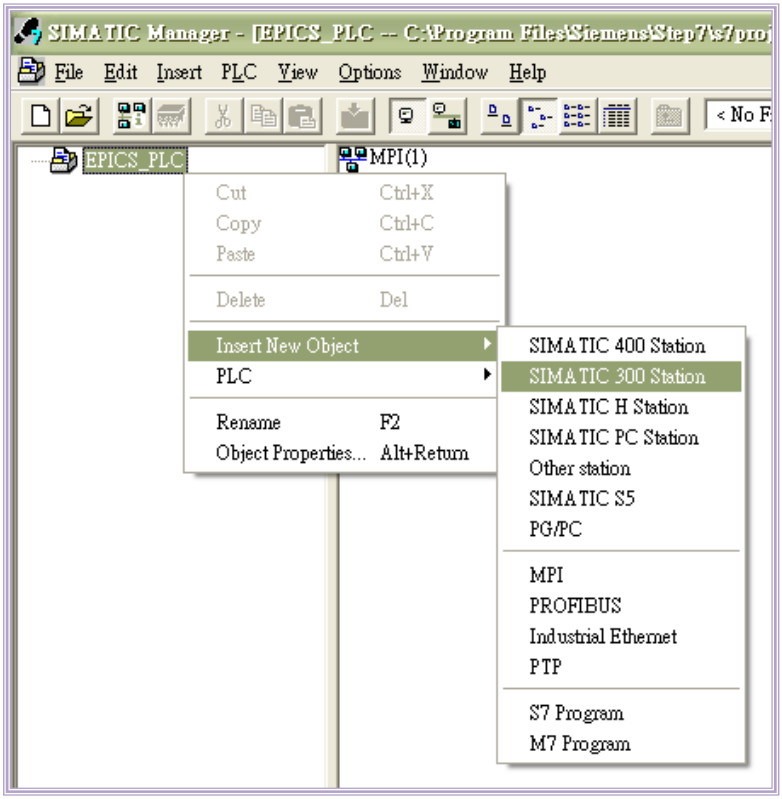


Figure 2

Step 2: Insert a PLC station in project.

On the "EPICS_PLC" project, click the right button to insert a SIMATIC Station as shown in Figure 2. The result is shown in Figure 3.

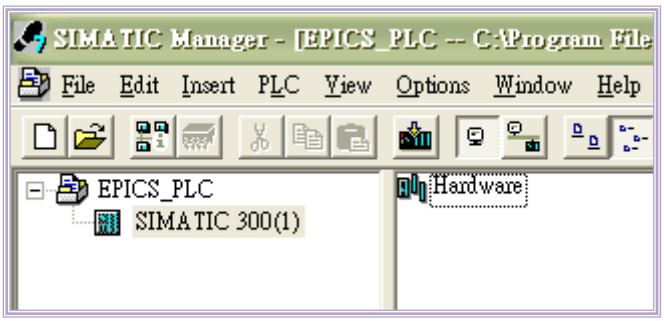


Figure 3

National Synchrotron Radiation Research Center,
RF Group, Yu-Han Lin

Step 3: Configure the hardware configuration of the PLC.

In the SIMATIC Station as shown in Figure 3, double-click “Hardware”; this operation leads you into the hardware configuration tool (figure 4). According to your PLC hardware, configure the modules at the right position. As all configurations are finished as shown in Figure 4, press button 1 to compile the configuration code, and then press button 2 to download the configurations into the PLC.

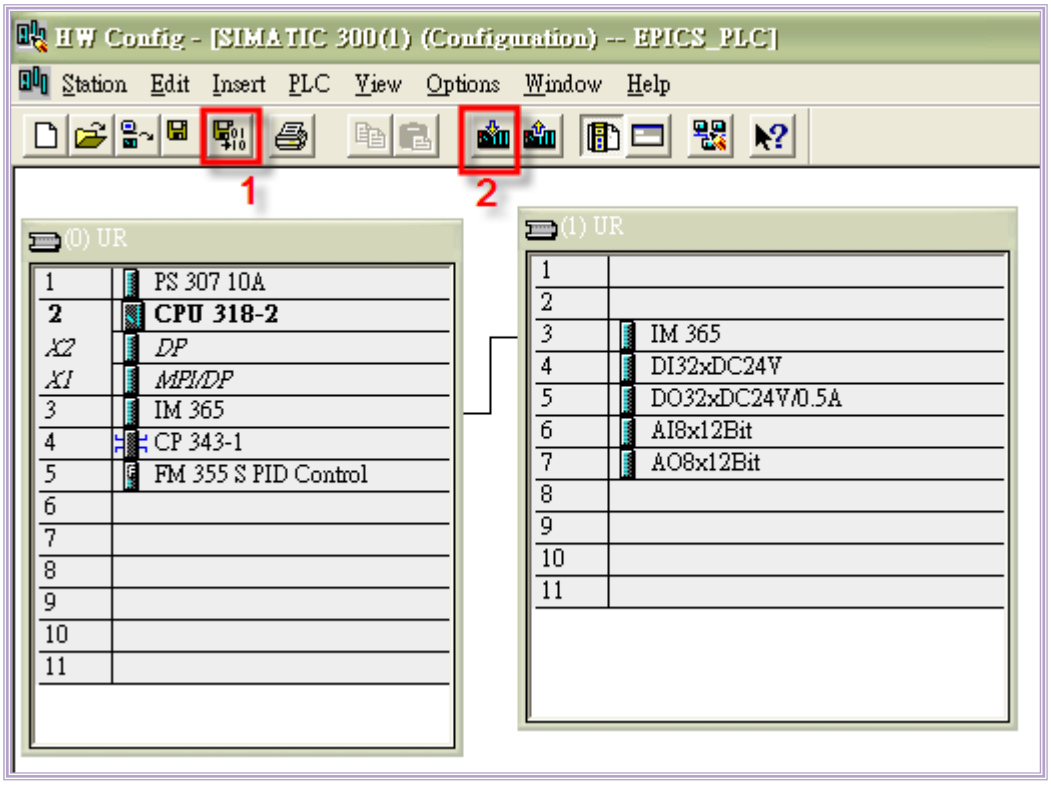


Figure 4

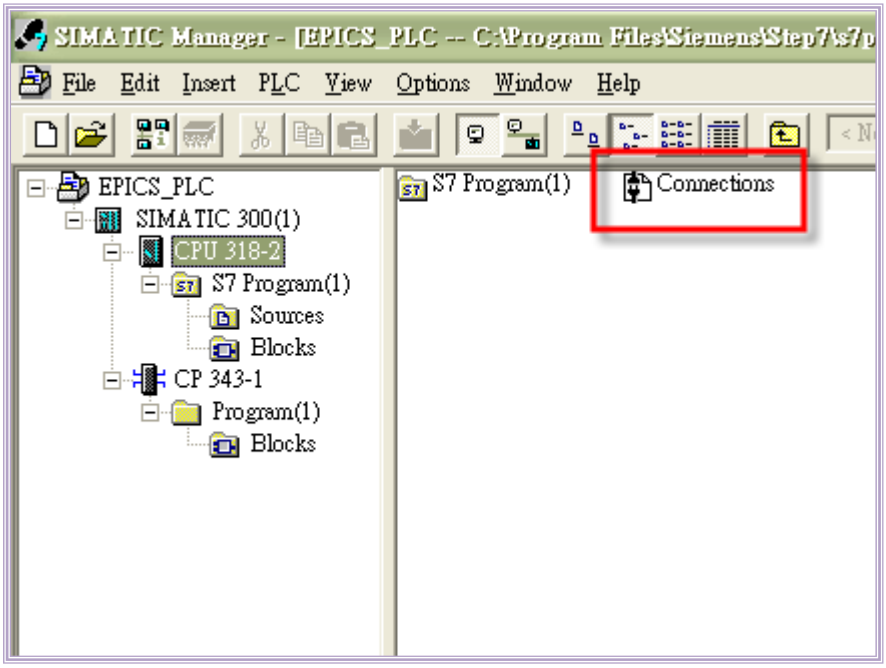


Figure 5

Step 4: Configure the connection profile of the PLC.

When the hardware is properly set up, the sections “CPU 318-2” and “CP 343-1” are automatically created in the SIMATIC Station as shown in Figure 5. In the “CPU 318-2” section, double-click the “Connection” to configure the PLC connections.

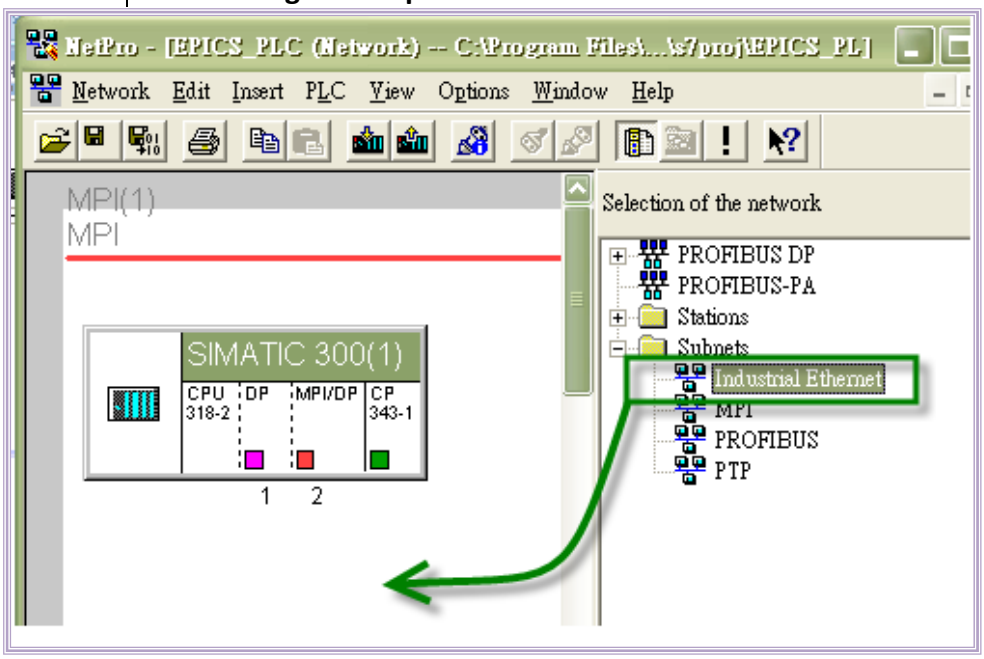


Figure 6

The network configuration is similar to the hardware configuration. You need only to drag a station or subnet into the project, and the entry becomes automatically created.

Here we seek to create an Industrial Ethernet subnet; so we drag the "Industrial Ethernet" into the project as shown in Figure 6. Figure 7 shows the result.

Double-click the "CP 343-1" module (the red rectangular area); a property configuration window pops up (Figure 8) to configure the Ethernet property such as the IP address and the mask.

For example, the ip address of the "CP 343-1" module is set to be 140.110.205.61.

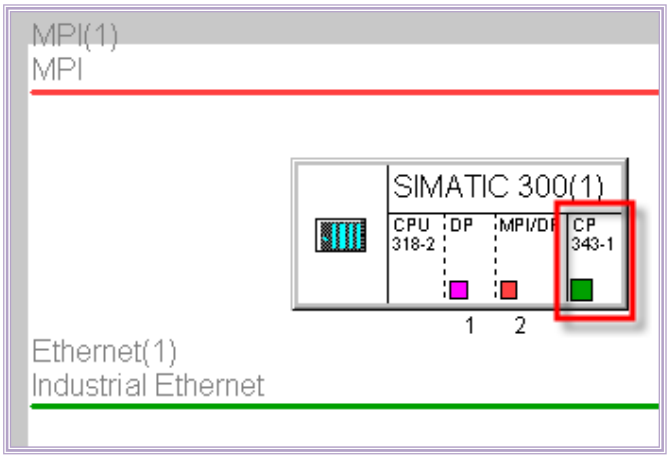


Figure 7

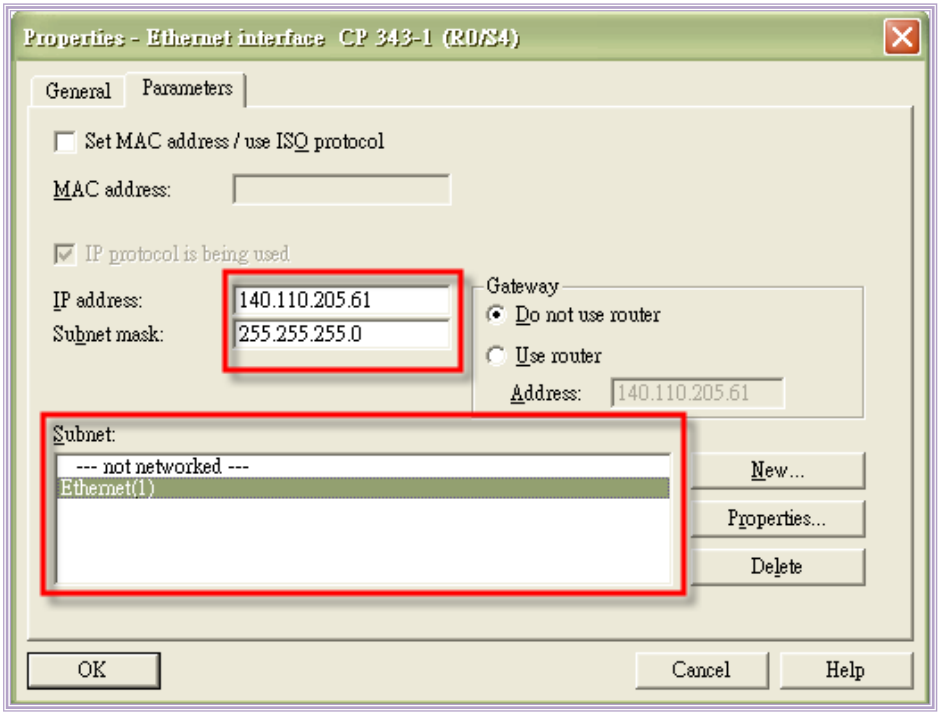


Figure 8

Select the "Ethernet(1)" in a subnet that we just created, so that the CP343-1 module will connect to Ethernet. Figure 9 shows the results of the configuration of CP343-1

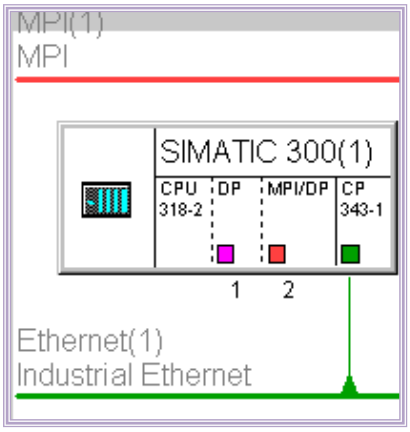


Figure 9

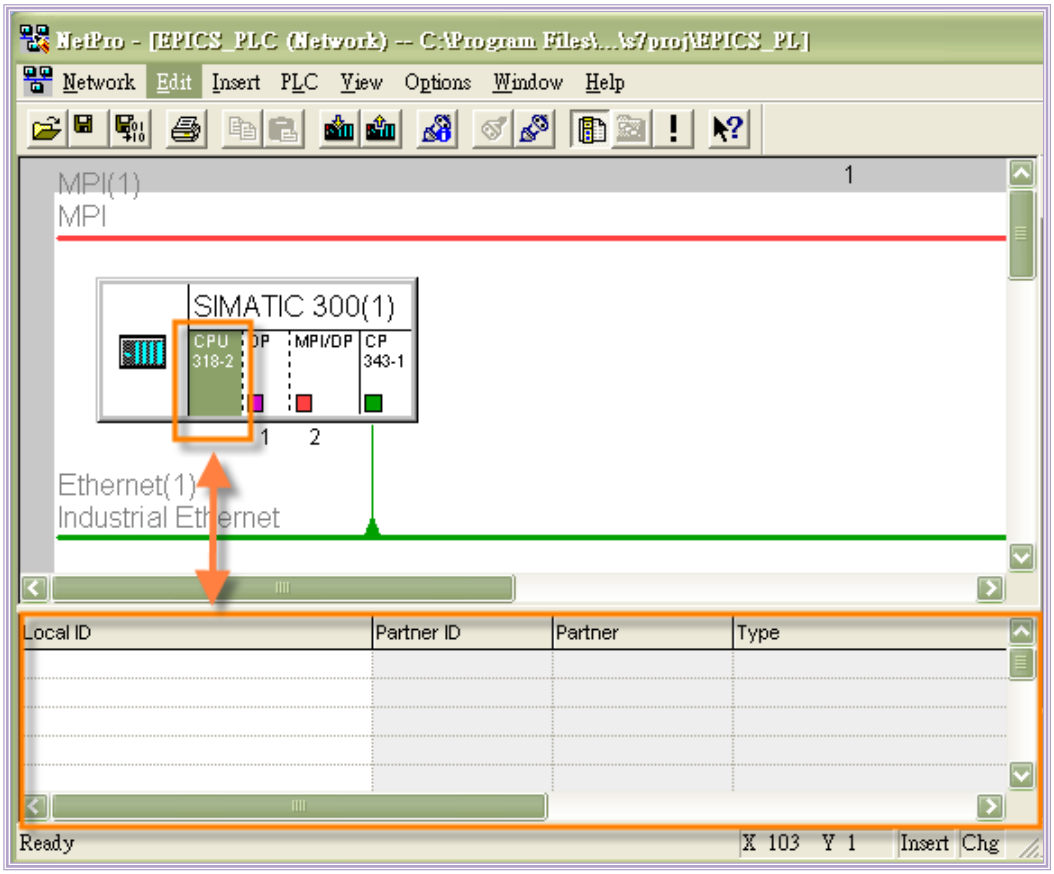


Figure 10

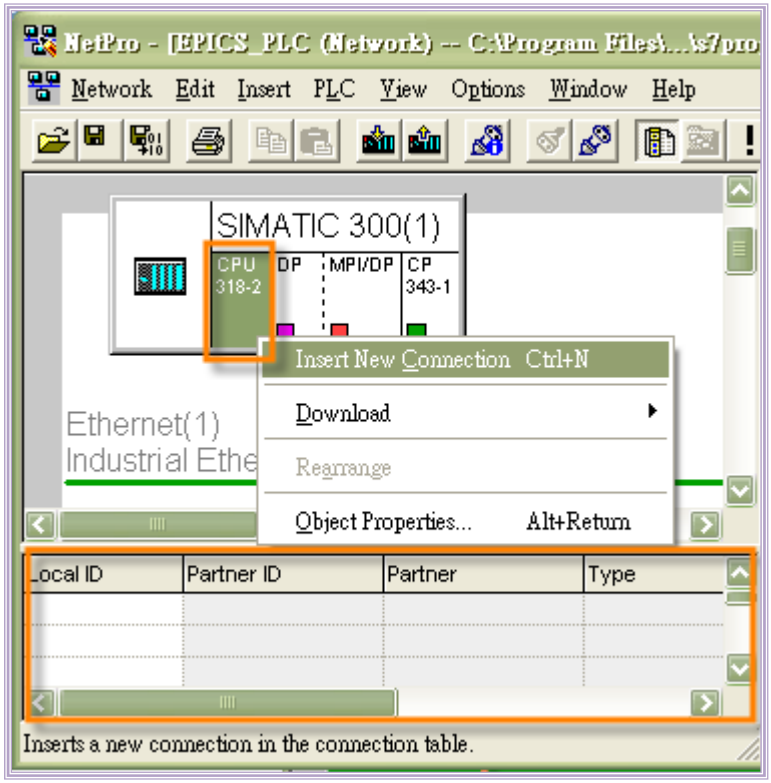


Figure 11

Step 5: Create a communication channel for EPICS.

There is no communication channel that can serve for data transmission / receiving. Select the CPU module; the communication channel becomes displayed at the bottom of the Netpro window.

To create a new communication channel, click the right button on the CPU module, and select “Insert New Connection”. As illustrated in Figure 12, a configuration window pops up.

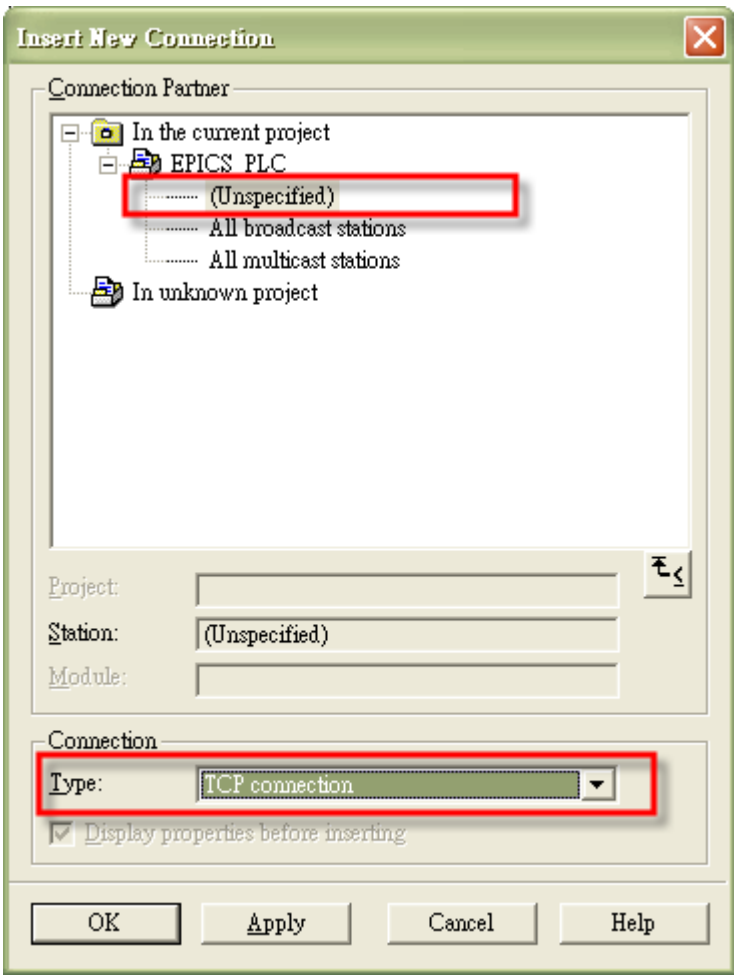


Figure 12

Use the "Connection Partner" group to specify the system or device with which PLC is going to communicate. For PLC to communicate with EPICS IOC, choose "Unspecified".

The "Connection" group is used to specify the communication protocol of PLC. Choose "TCP connection" for EPICS IOC communication, then press the "OK" button. A notification window will appear as shown in Figure 13; press "OK" again to ignore the message.

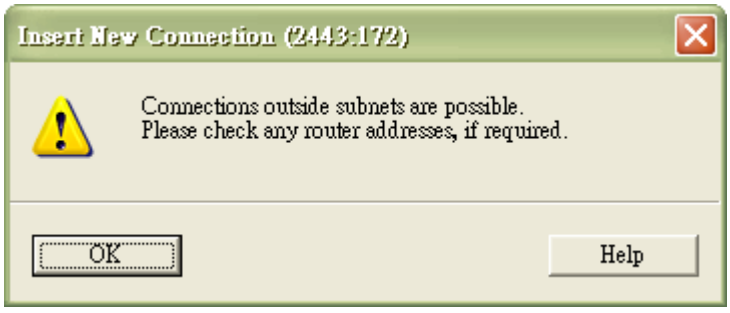


Figure 13

The configuration program leads you to configure the property of the "TCP connection". As demonstrated in Figure 14, the "General Information" tab shows the ID and LADDR parameters. These two parameters are the identification number of this communication channel, and are used later.

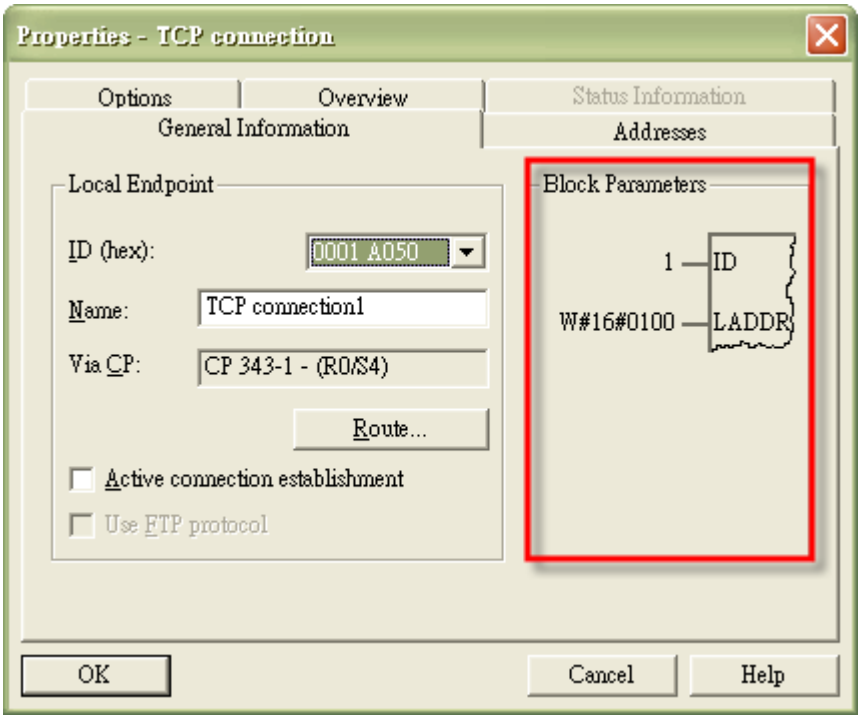


Figure 14

The “Addresses” tab is used to specify the ip and port parameters of the local (PLC) and the partner (IOC). The local ip address has been set at step 4 (Figure 8). The default local port is 2000.

Do NOT specify the ip address and port of partner. If these two parameters were specified, the communication between PLC and IOC might fail.

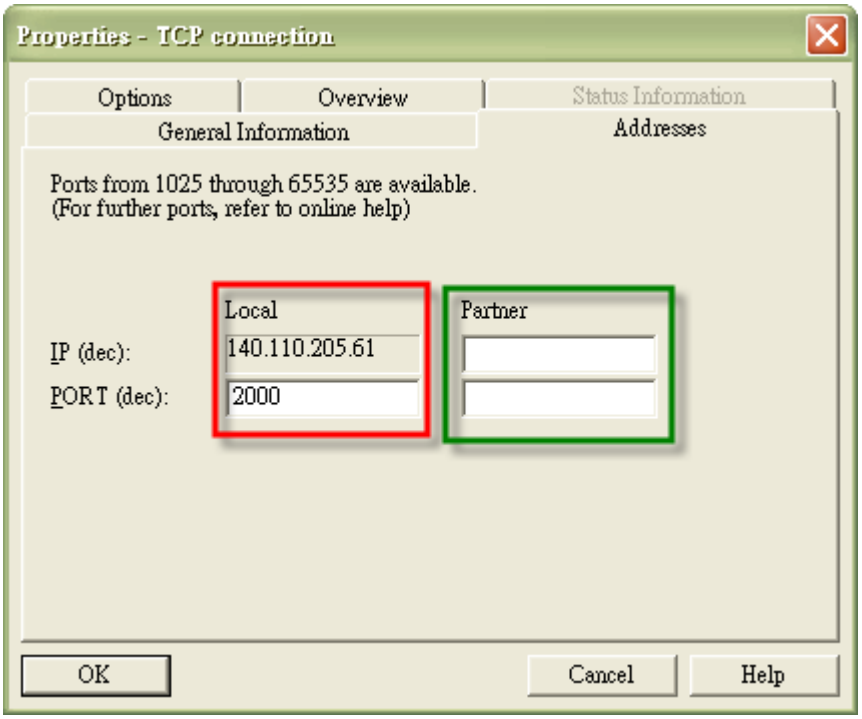
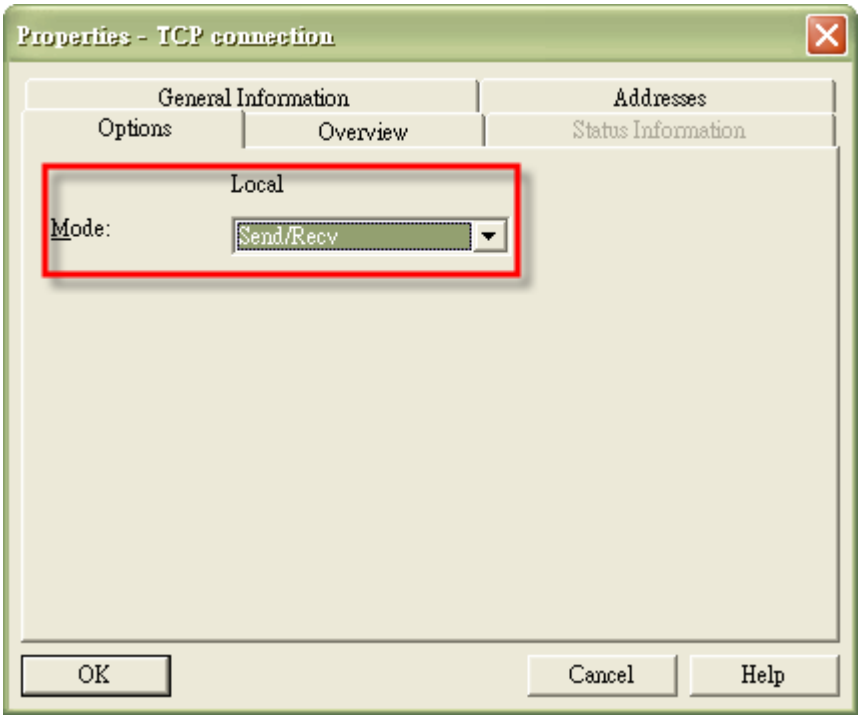


Figure 15



The “Options” tab is used to specify the communication mode of PLC. Set the mode to be “Send/Recv”.

Press “OK”: the configuration of TCP connection is then finished.

Figure 16

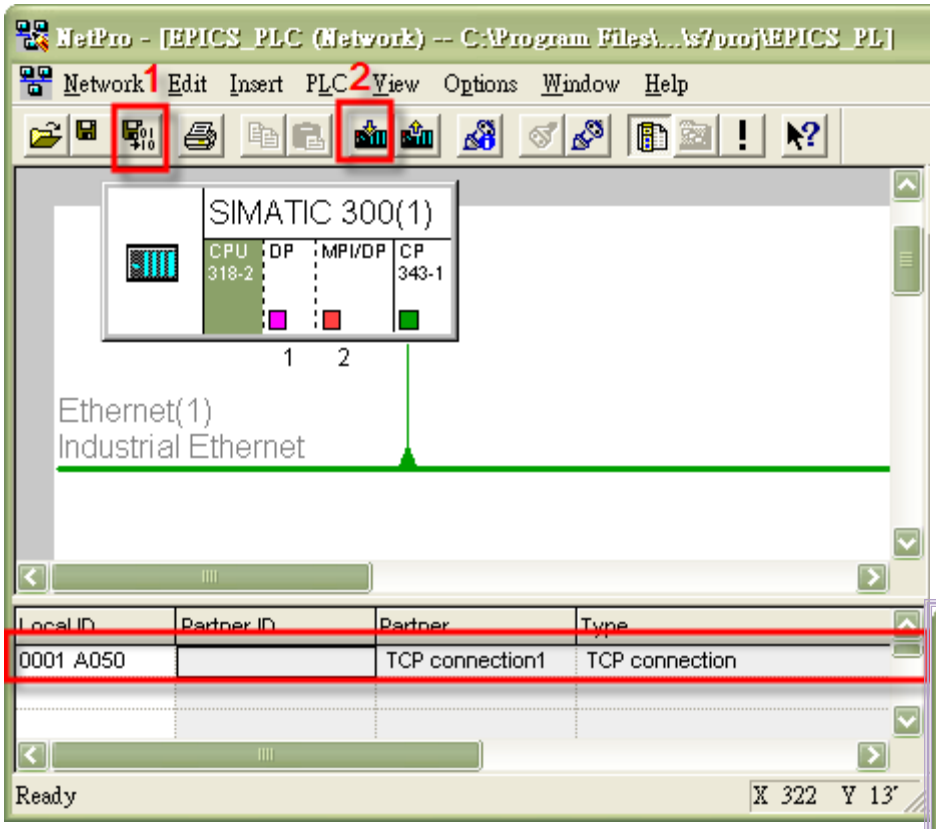


Figure 17

When the connection configuration is finished, as shown in Figure 17, click on the CPU module; the connection information becomes displayed at the bottom of the NetPro window.

Click button 1 to compile the network configurations, and then click button 2 to download the setting into the PLC CPU.

If all configurations are correct, a "no error" message appears on the screen.

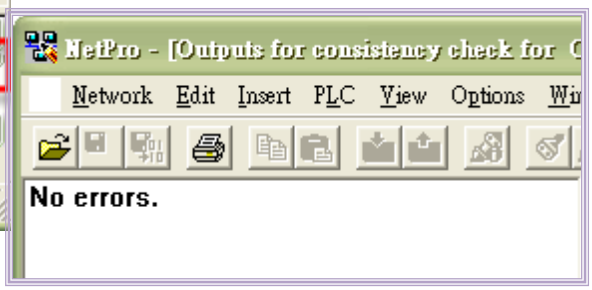


Figure 18

Step 6: Insert data blocks for transmission and receiving.

Back to the SIMATIC Manager, select the "Blocks" section in the tree. There is only one organization block "OB1" in the "Block" section as illustrated in Figure 19.

We need to create two data blocks: one serves for data transmission, and the other for data receiving.

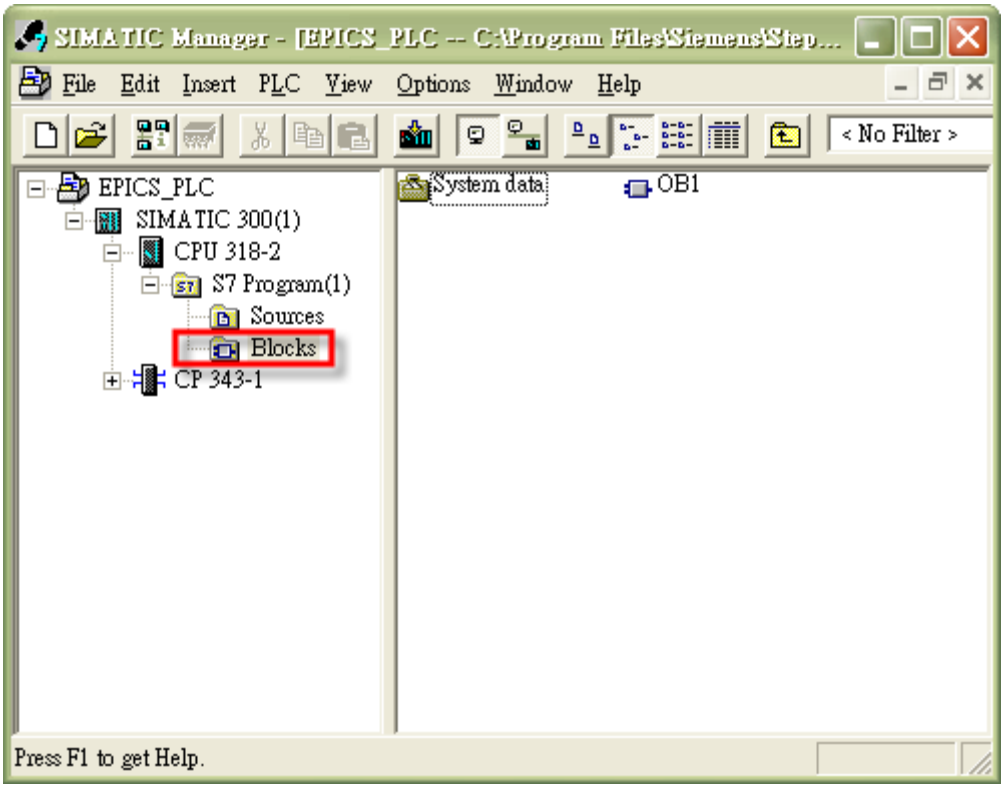
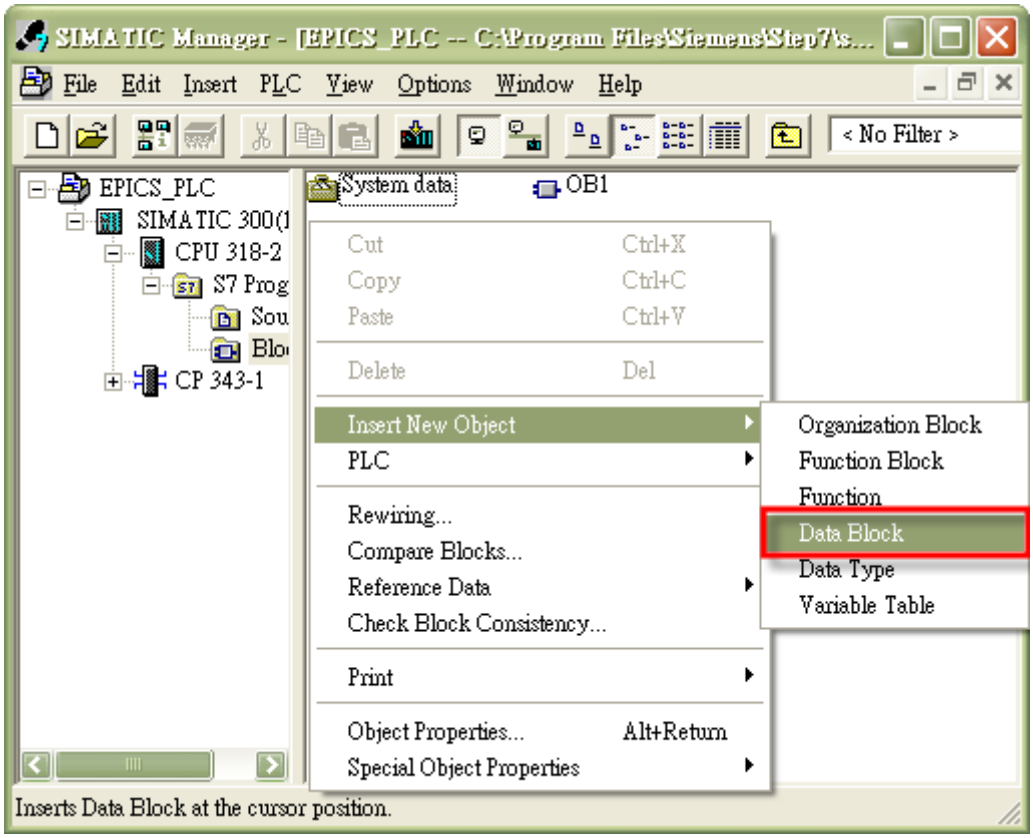


Figure 19



Click the right button on the screen; select "Insert New Object" -> "Data Block" as shown in Figure 20.

Figure 20

In the Data Block properties configuration window (Figure 21), rename the data block "DB10", and set the type to be "Shared DB".

Create another data block, and rename it "DB11" with type "Shared DB".

We will later use "DB10" for data transmission and "DB11" for data receiving.

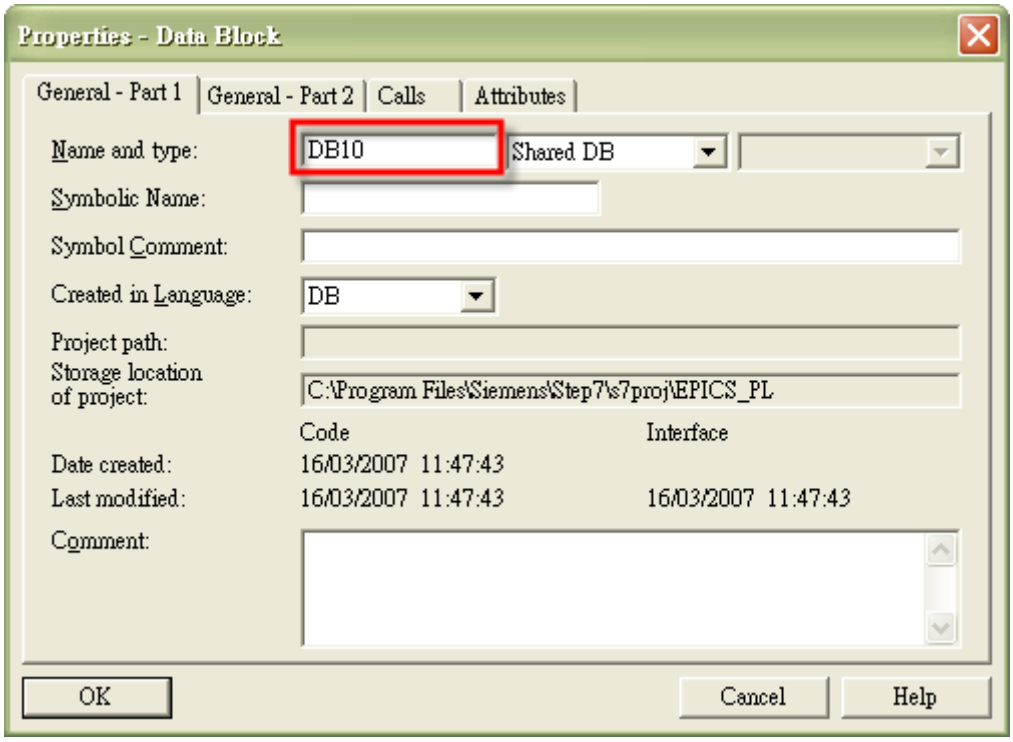


Figure 21

As shown in Figure 22, two data blocks were created, but the contents of these two data blocks are not yet defined. We therefore need to define the contents of these two data blocks.

Double-click on "DB10"; a configuration window as Figure 23 pops up. Modify the contents of DB10, as shown in Figure 24.

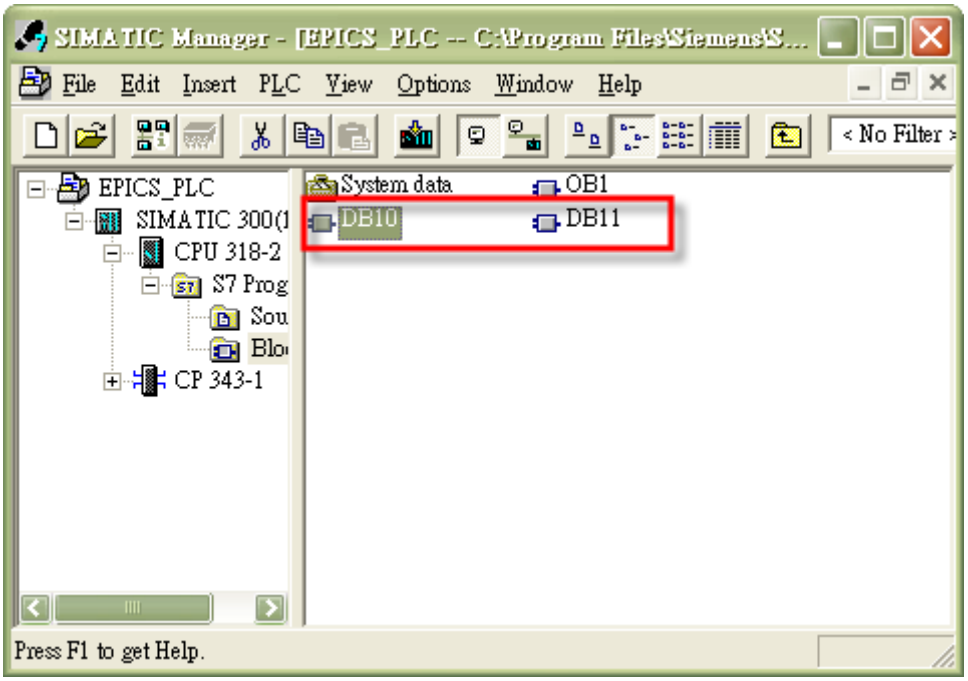


Figure 22

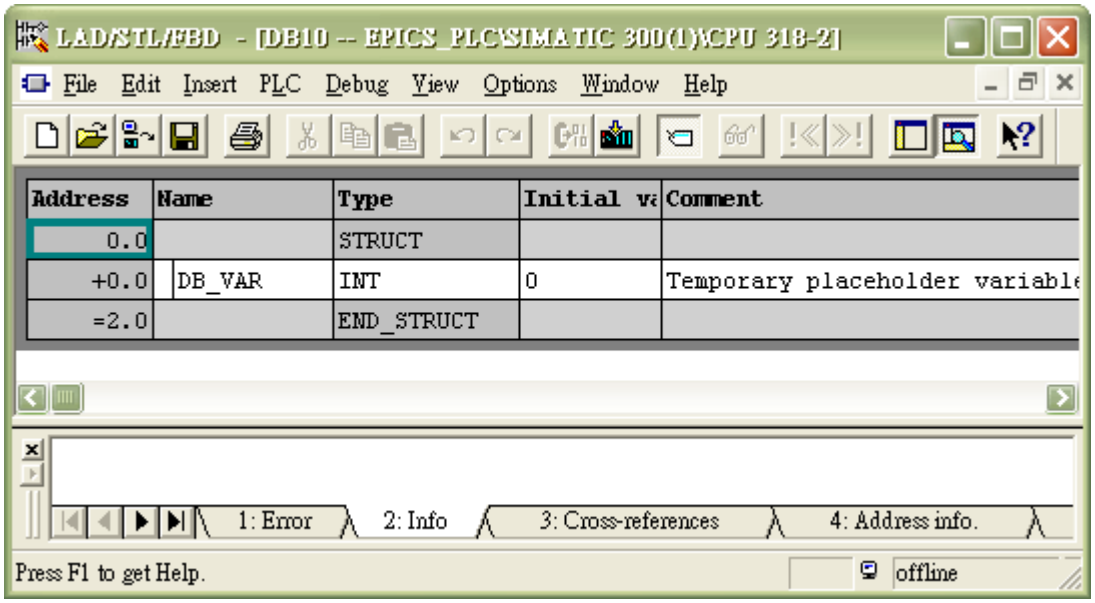


Figure 23

Open "DB11", and modify the content of DB11 to be the same as DB10.

Save these modifications, and close the configuration window.

Note that you can define the contents of DB10 and DB11 to meet your data format that you seek to transmit and to receive. Figure 24 is just illustrated as an example.

Address	Name	Type	Initial va	Comment
*0.0		STRUCT		
+0.0	Contents	ARRAY[0..119]		Temporary placeholder vari
*2.0		WORD		
=240.0		END_STRUCT		

Figure 24

Step 7: Insert functions for data transmission and receiving.

In the SIMATIC Manager, click the right button on the screen and select "Insert New Object" -> "Function". A configuration window for a function pops up.

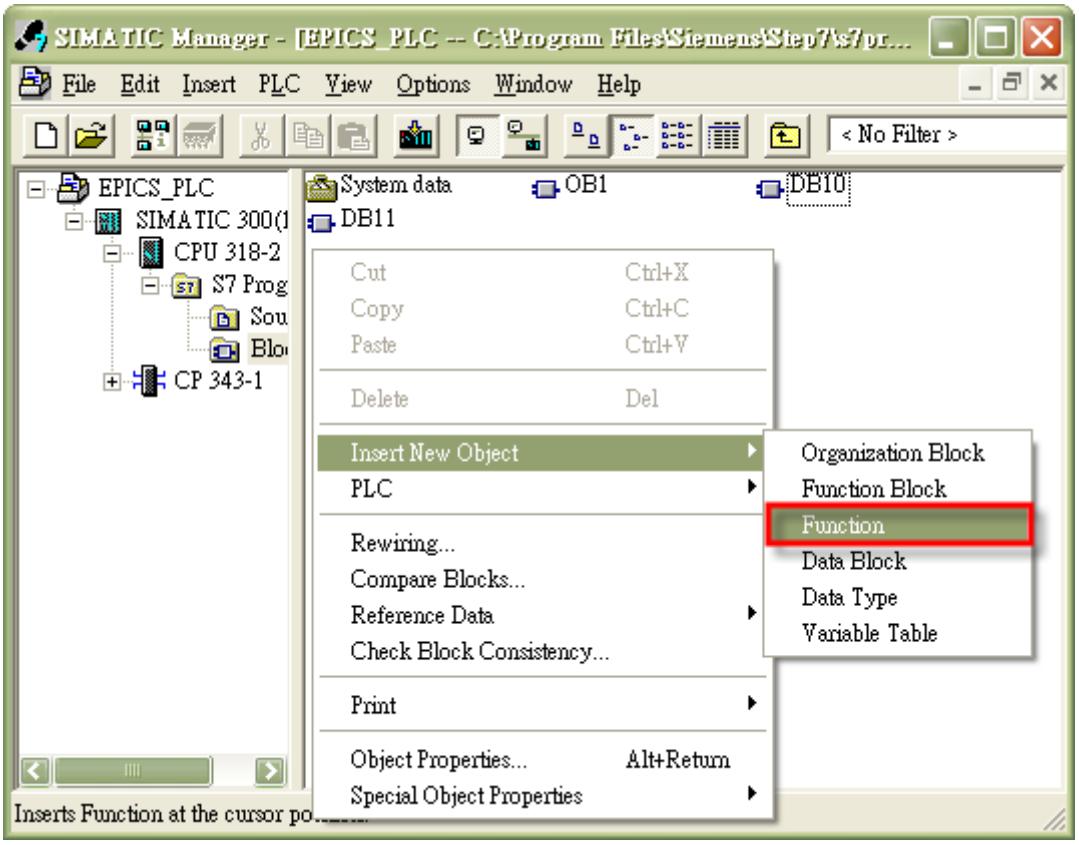


Figure 25

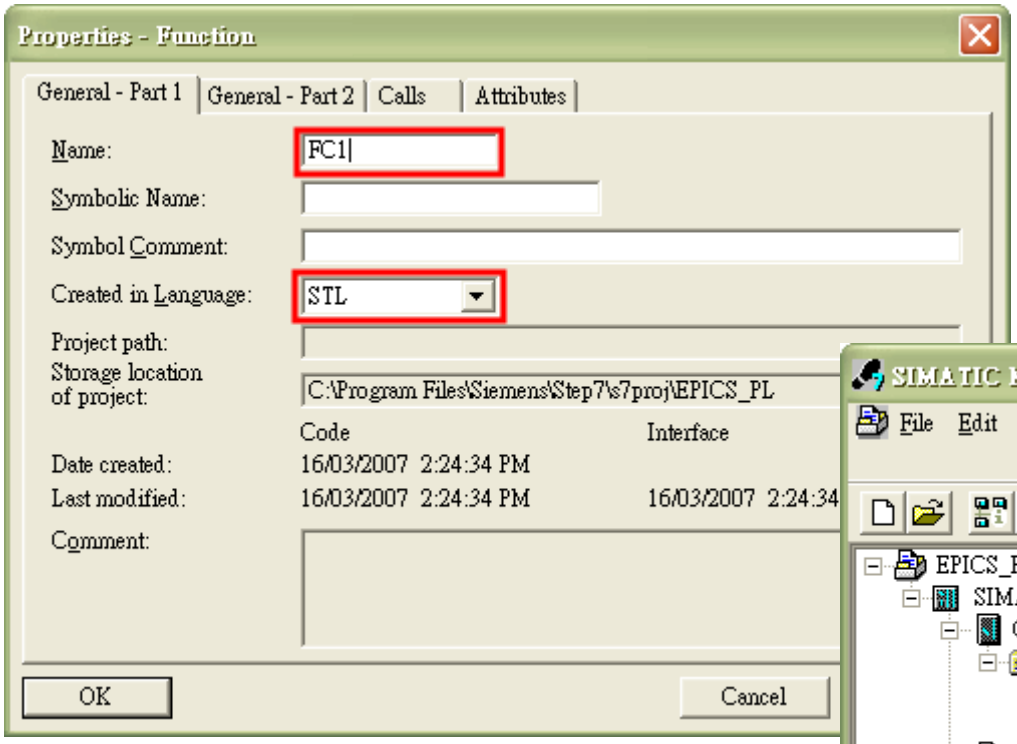


Figure 26

Rename this function to be "FC1", and you can choose the programming language in the "Created in Language" row.

Return to the SIMATIC Manager, and double-click "FC1" to load "FC1" for editing.

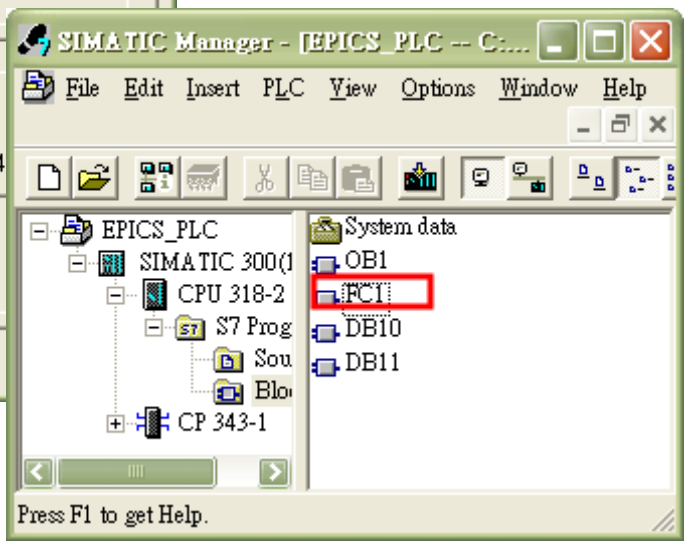


Figure 27

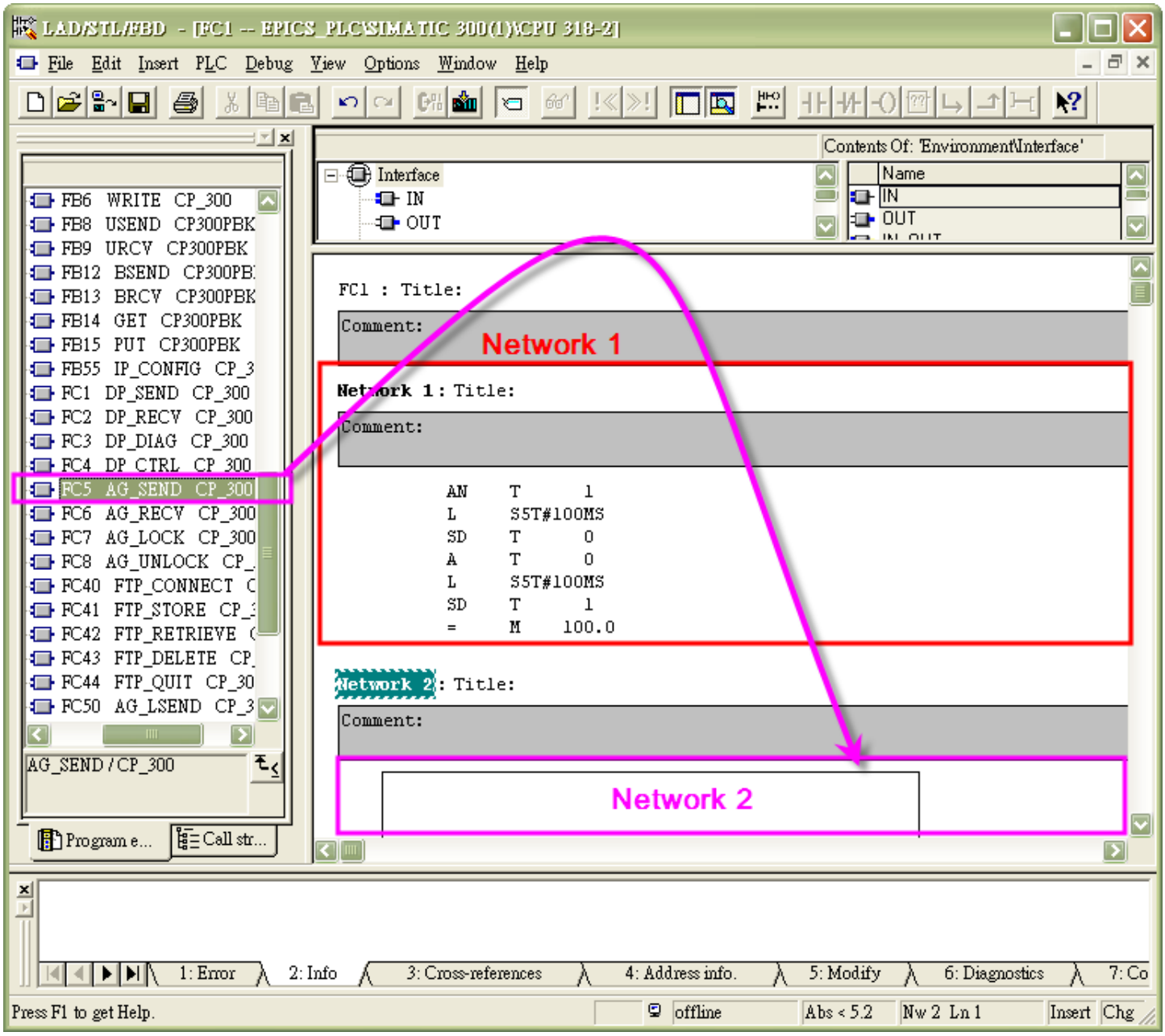


Figure 28

Figure 28 shows the programming editor of FC1. In network 1, the memory address M100.0 is turned on/off per 100 ms. We can use this memory address as the switch to turn on/off the data transmission.

In network 2, we need to call the sub-program "AG_SEND", which can be found in the "Program elements" tree. Note that the "AG_SEND" function depends on the module.

This function is used to send data. Double-click the "AG_SEND"; it becomes automatically added to the selected network.

Table 1

```
Network 1:

AN   T    1
L    S5T#100MS
SD   T    0
A    T    0
L    S5T#100MS
SD   T    1
=    M    100.0
```

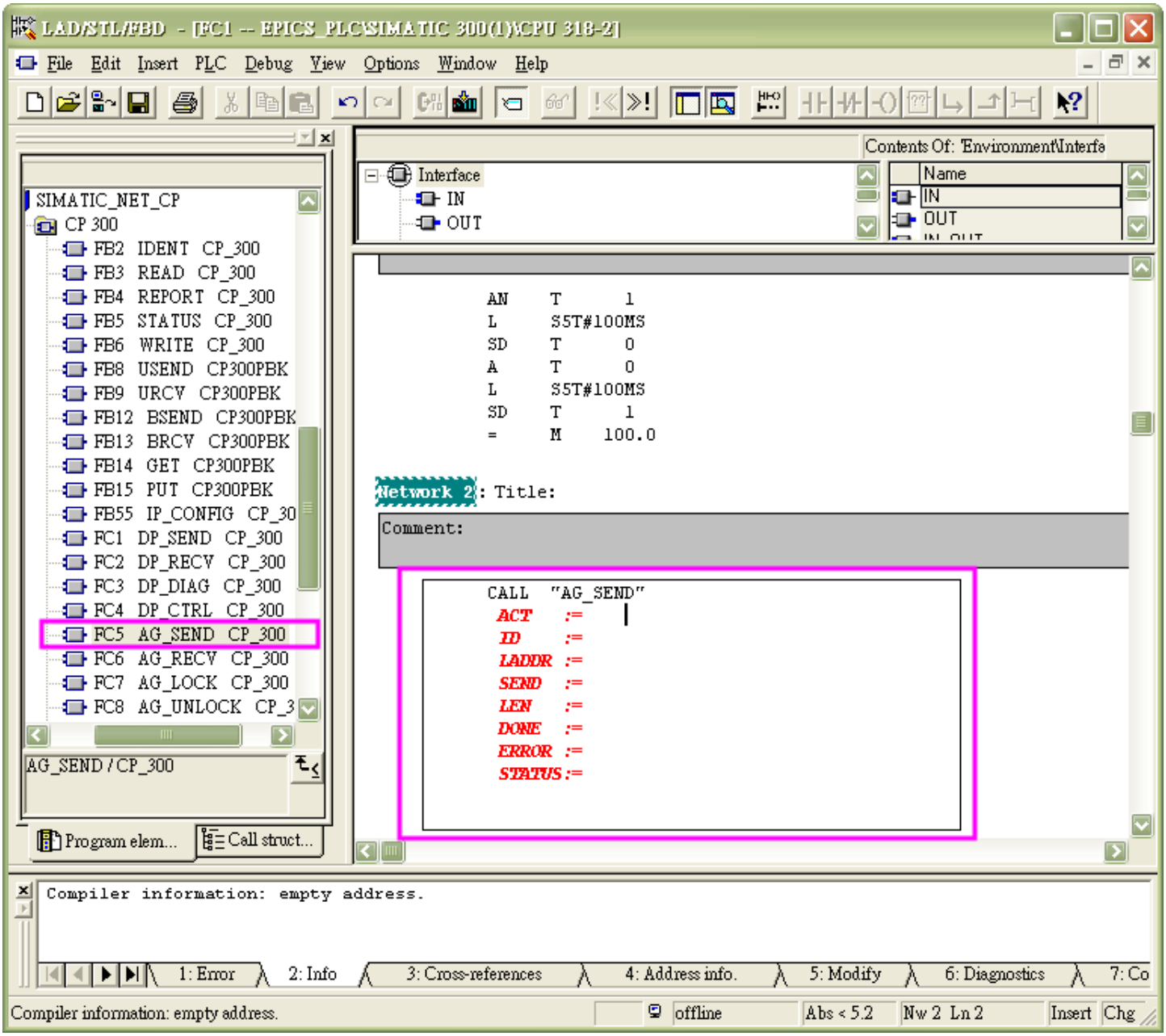


Figure 29

The editor will ask you to input the required parameters. Fill these rows with proper parameters, then save and quit.

Note:

1) The "ID" and "LADDR" are used to identify the connection channel; they must match the values in Figure 14.
2) For CP300 series, the maximum length of "AG_SEND" is bounded by 240 bytes.
3) In network 2, we use memory address M100.0 to turn on/off the transmission.
4) Data block (DB10) of length 240 bytes is sent.

Table 2

```
Network 2:

CALL    "AG_SEND"
ACT     :=M100.0
ID      :=1
LADDR   :=W#16#100
SEND    :=DB10.Contents
LEN     :=240
DONE    :=M0.0
ERROR   :=M0.1
STATUS  :=MW10
```

Back at the SIMATIC Manager, “FC5” (AG_SEND) is automatically inserted into this project.

Double-click “OB1” to load “OB1” for editing.

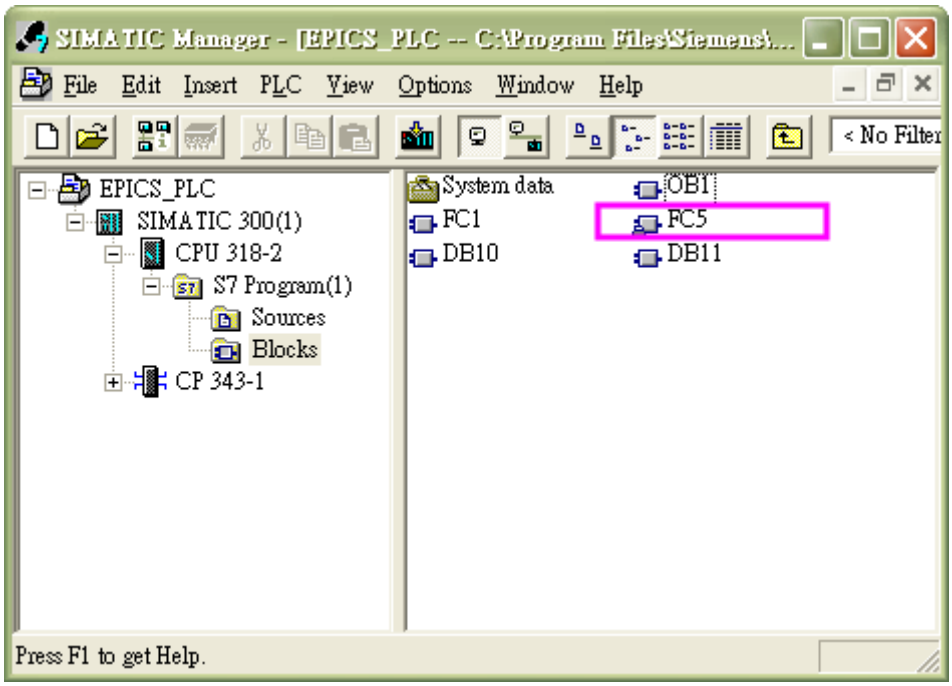


Figure 30

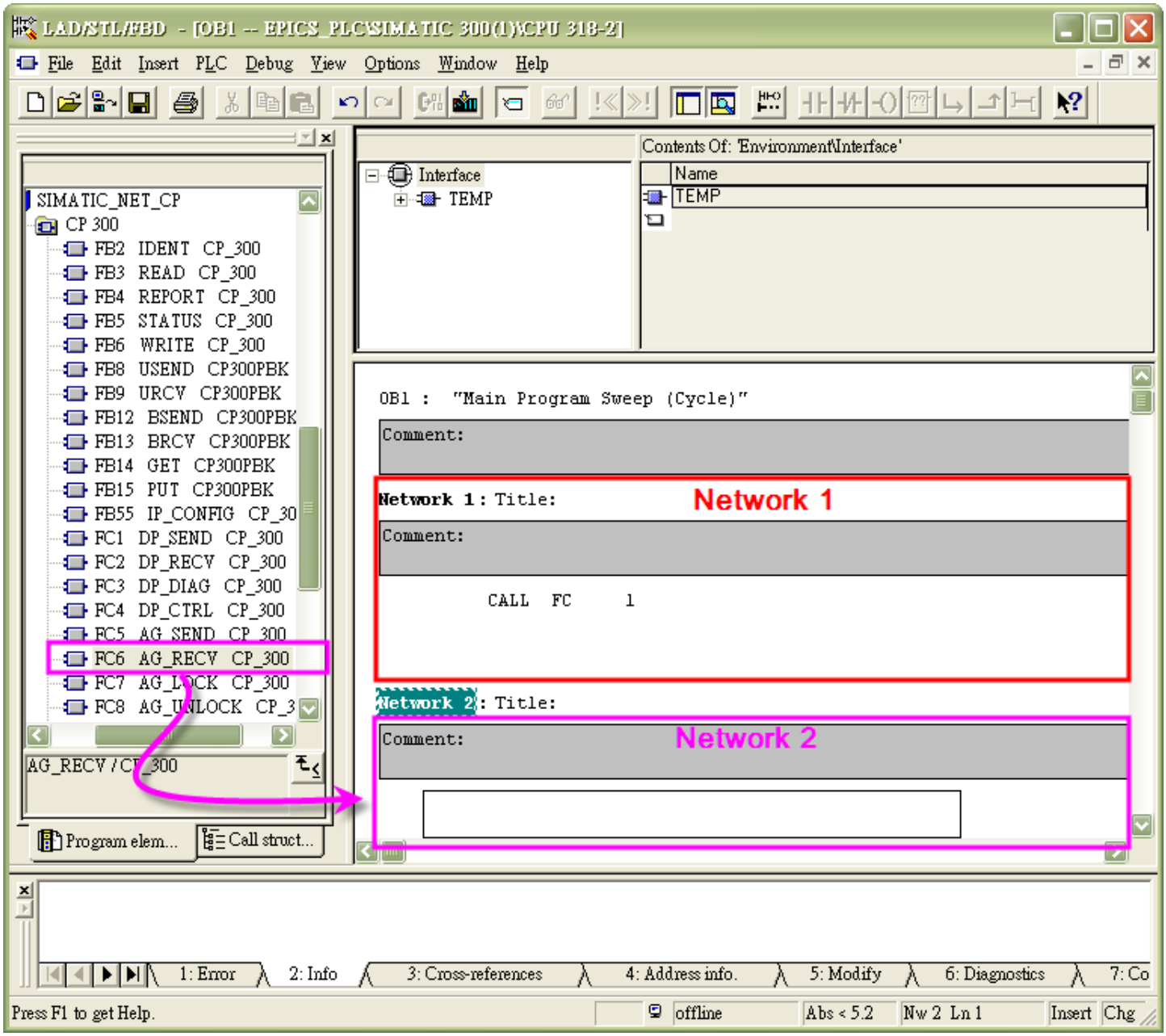


Figure 31

The editor of OB1 is shown in Figure 31.

The “FC1”, which is created to send data, is called in network 1. To receive data from IOC, we need an “AG_RECV” function to listen to the messages from Ethernet.

Again, move the mouse to network 2, and double-click the “AG_RECV” function. The function becomes automatically inserted into “OB1” and the project.

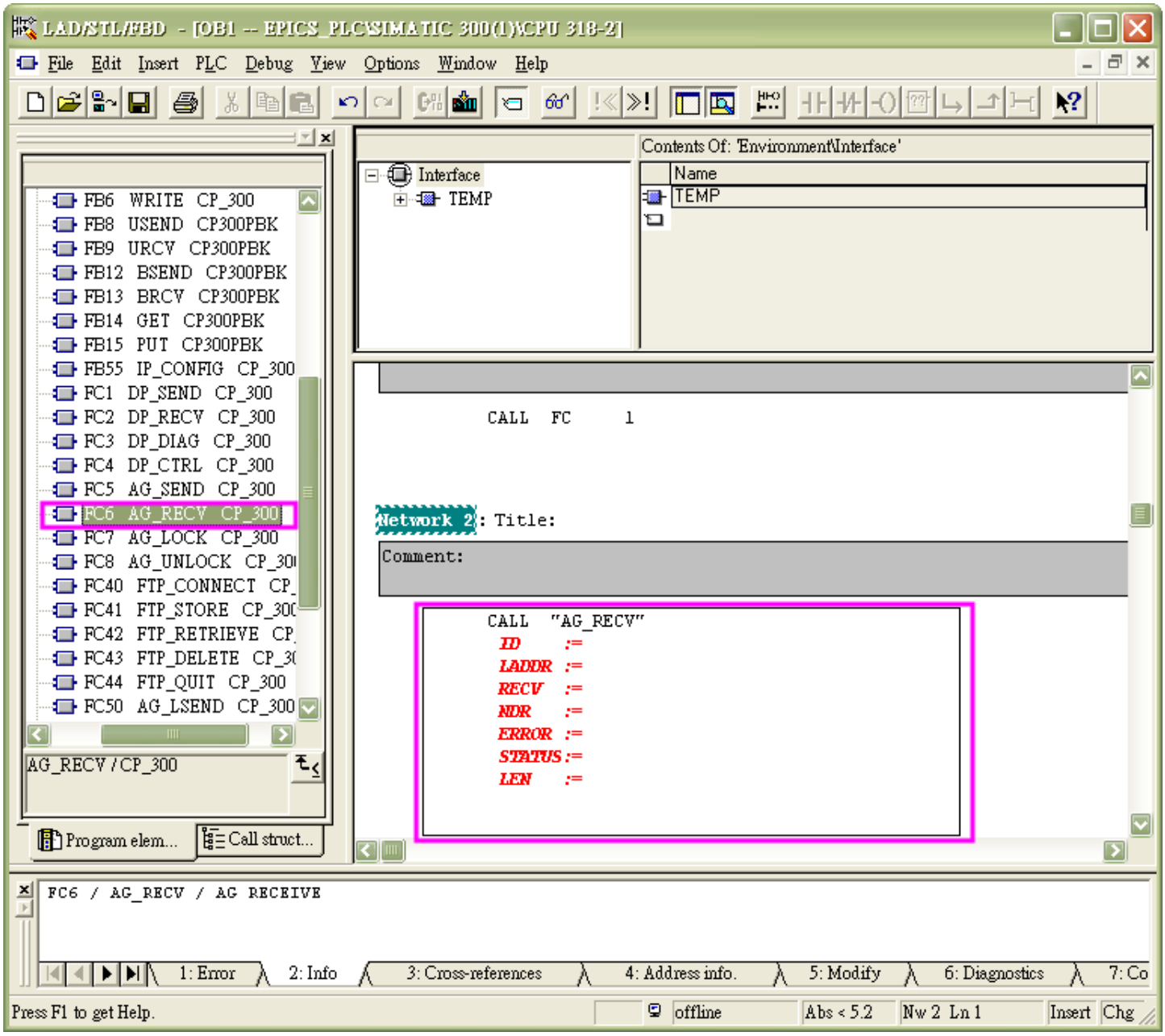


Figure 32

Again, the editor asks you to fill in the required parameters. Fill these rows with proper parameters, then save and quit.

Note:

1) The "ID" and "LADDR" must also match the values in Figure 14.
2) For the CP300 series, the maximum length of "AG_RECV" is also bounded by 240 bytes.
3) The received data will be written into DB11, and the length of the received message will be written to MW30.

Table 3

Network 2:

```
CALL    "AG_RECV"
ID      :=1
LADDR   :=W#16#100
RECV    :=P#DB11.DBX 0.0 BYTE 240
NDR     :=M1.2
ERROR   :=M1.3
STATUS  :=MW20
LEN     :=MW30
```

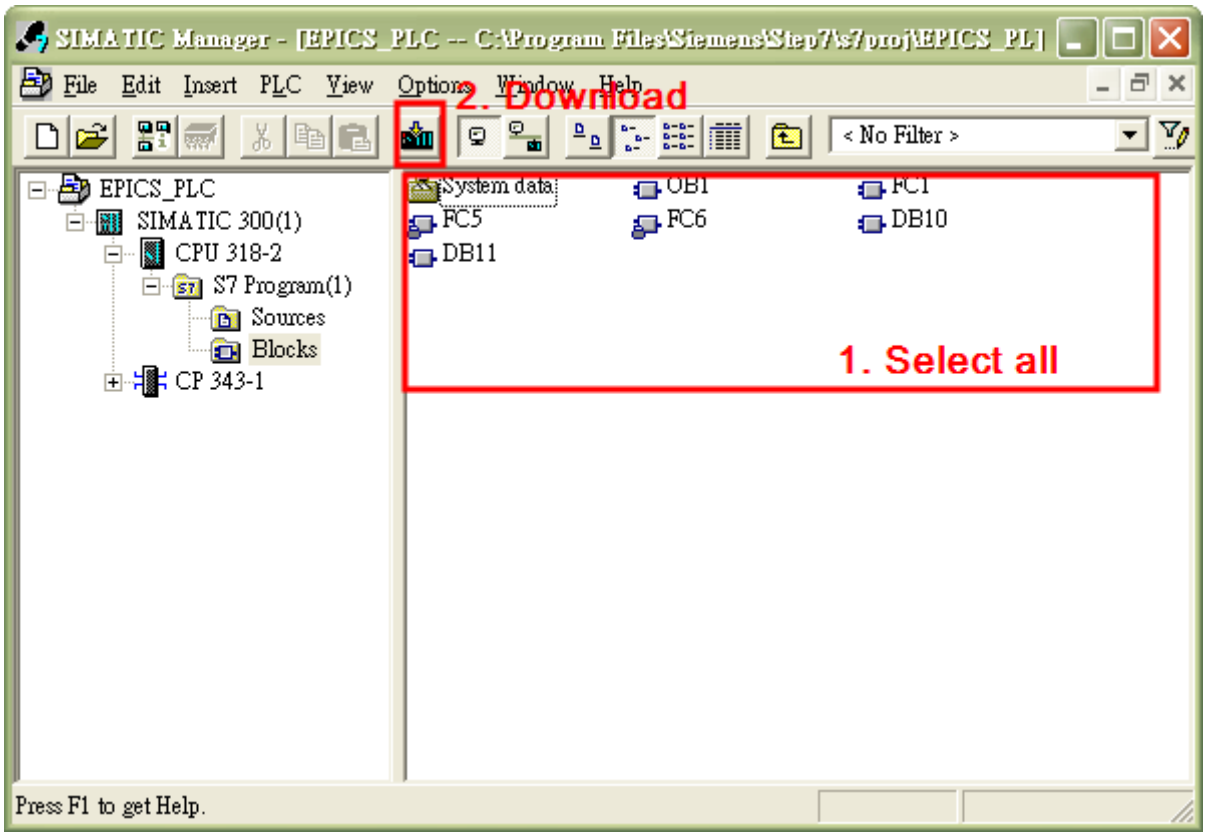


Figure 33

Select all components in SIMATIC Manager, and press the download button to download all components into PLC. The general-purpose communication configuration is then finished.

Communication Example

In this example, we will
1) confirm the connection between PLC and EPICS IOC,
2) put an analog value into PLC from IOC,
3) multiply this analog value by 2 in PLC,
4) read this value,
5) read a bit (switch) status, and
6) modify this bit status (turn on/off the switch).

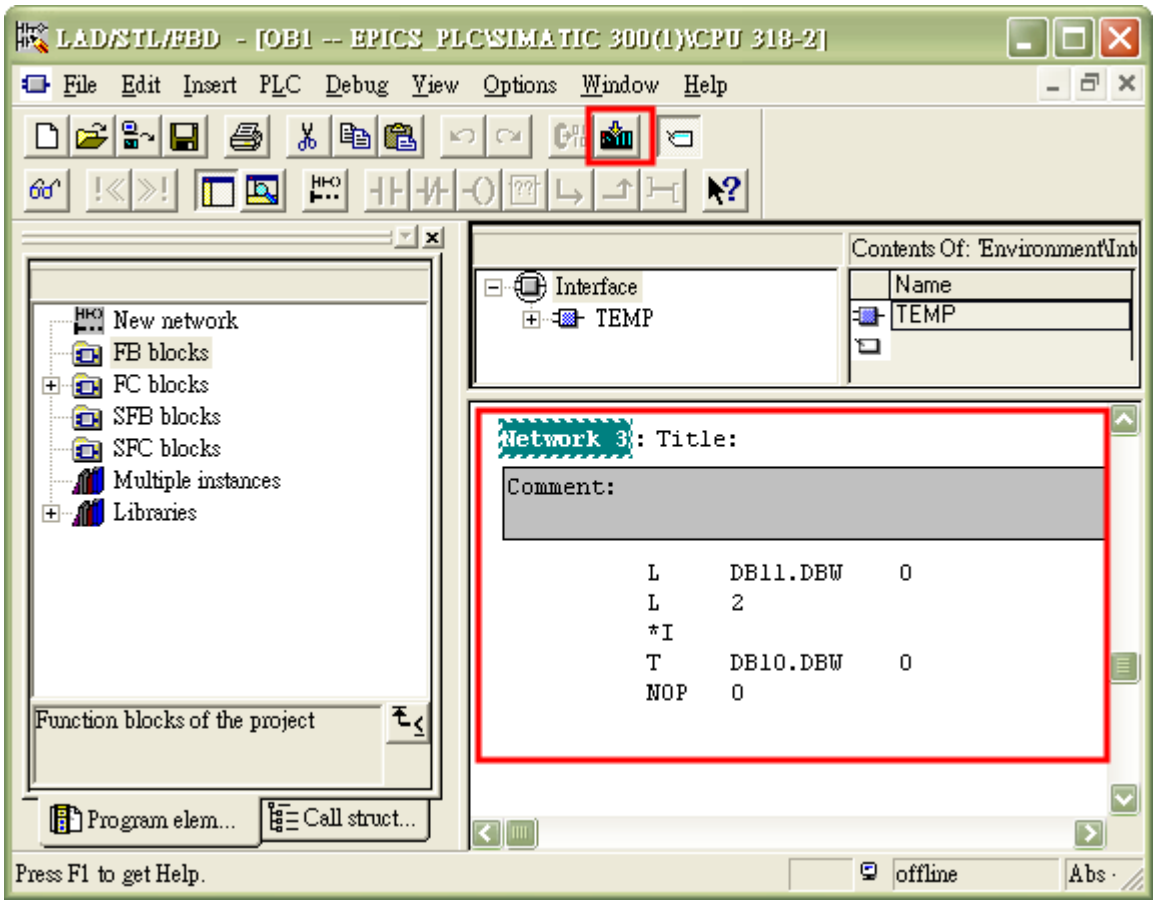


Figure 34

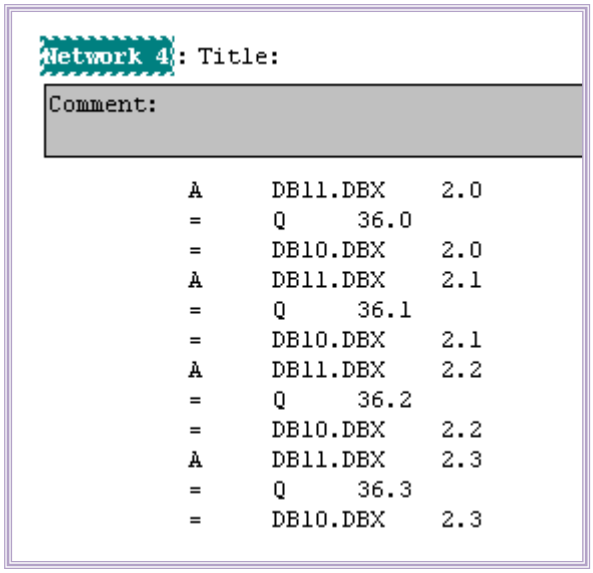


Figure 35

In the PLC terminal, insert a new network (Network 3) in Organization block 1 (OB1) to
1) load the value from DB11.DBW0,
2) multiply this value by 2, and
3) save the value to DB10.DBW0.

Insert another new network (Network 4) to read the bit status of DB11 and turn on/off the output device. The program is shown in Figure 35.

Remember to download the modified OB1 into PLC by pressing the download button. Then switch the PLC to RUN mode.

Table 4 shows the modified contents of data base “example.db”.

The record “s7-status” is used to verify the PLC connection status. The record “ai16-1” is an analog input record that reads the value of the first word of DB10 (i.e. DB10.DBW0). The record “ao16-1” is an analog output record that writes the value to the first word of DB11 (i.e. DB11.DBW0). The records “bi-x/bo-x” are the binary input/output records; they read/write the value from/to the x-th bit of the third byte of DB10/DB11.

Table 4

```
record(bi, s7-status) {
  field(DTYP, "S7plc stat")
  field(INP, "@ Station:0")
  field(SCAN, "I/O Intr")
  field(ZNAM, "disconnected")
  field(ONAM, "connected")
  field(ZSV, "MAJOR")
  field(FLNK, "s7-status-counter")
}
record(calc, s7-status-counter) {
  field(INPA, "s7-status-counter")
  field(CALC, "A+1")
  field(FLNK, "s7-disconnect-counter")
}
record(calc, s7-disconnect-counter) {
  field(INPA, "s7-status")
  field(INPB, "s7-disconnect-counter.LA")
  field(INPC, "s7-disconnect-counter")
  field(CALC, "(A=0&&B=1)?C+1:C")
}
record(ai, ai16-1) {
  field(SCAN, " I/O Intr")
  field(DTYP, "S7plc")
  field(INP, "@ Station:0/0 T=WORD")
}
record(ao, ao16-1) {
  field(DTYP, "S7plc")
  field(OUT, "@ Station:0/0 T=WORD")
}
```

```
record(bi, bi-1) {
  field(SCAN, "I/O Intr")
  field(DTYP, "S7plc")
  field(INP, "@ Station:0/2 B=0 T=BYTE")
}
record(bi, bi-2) {
  field(SCAN, "I/O Intr")
  field(DTYP, "S7plc")
  field(INP, "@ Station:0/2 B=1 T=BYTE")
}
record(bi, bi-3) {
  field(SCAN, "I/O Intr")
  field(DTYP, "S7plc")
  field(INP, "@ Station:0/2 B=2 T=BYTE")
}
record(bi, bi-4) {
  field(SCAN, "I/O Intr")
  field(DTYP, "S7plc")
  field(INP, "@ Station:0/2 B=3 T=BYTE")
}
record(bo, bo-1) {
  field(DTYP, "S7plc")
  field(OUT, "@ Station:0/2 B=0 T=BYTE")
  field(PINI, "YES")
}
record(bo, bo-2) {
  field(DTYP, "S7plc")
  field(OUT, "@ Station:0/2 B=1 T=BYTE")
  field(PINI, "YES")
}
record(bo, bo-3) {
  field(DTYP, "S7plc")
  field(OUT, "@ Station:0/2 B=2 T=BYTE")
  field(PINI, "YES")
}
record(bo, bo-4) {
  field(DTYP, "S7plc")
  field(OUT, "@ Station:0/2 B=3 T=BYTE")
  field(PINI, "YES")
}
```

Table 5 shows the startup script of the IOC. Note that the <IP address> and the <port> field must exactly match the PLC connection configurations (see Figure 15). Set the input size and output size to 240 bytes, which is the maximum limit of a CP343 module.

Table 5

```
dbLoadDatabase ../../dbd/s7plcApp.dbd
s7plcApp_registerRecordDeviceDriver
var s7plcDebug 1

#s7plcConfigure name,IPaddr,port,inSize,outSize,bigEndian,recvTimeout,sendIntervall
#connects to PLC <name> on address <IPaddr> port <port>
#<inSize>          :  size of data bock PLC -> IOC [bytes]
#<outSize>         :  size of data bock IOC -> PLC [bytes]
#<bigEndian>=1    :  Motorola format data (MSB first)
#<bigEndian>=0    :  Intel format data (LSB first)
#<recvTimeout>     :  time to wait for input before disconnecting [ms]
#<sendIntervall>   :  time to wait before sending new data to PLC [ms]

s7plcConfigure Station:0,140.110.205.61,2000,240,240,1,2000,500

dbLoadRecords example.db

iocInit
```

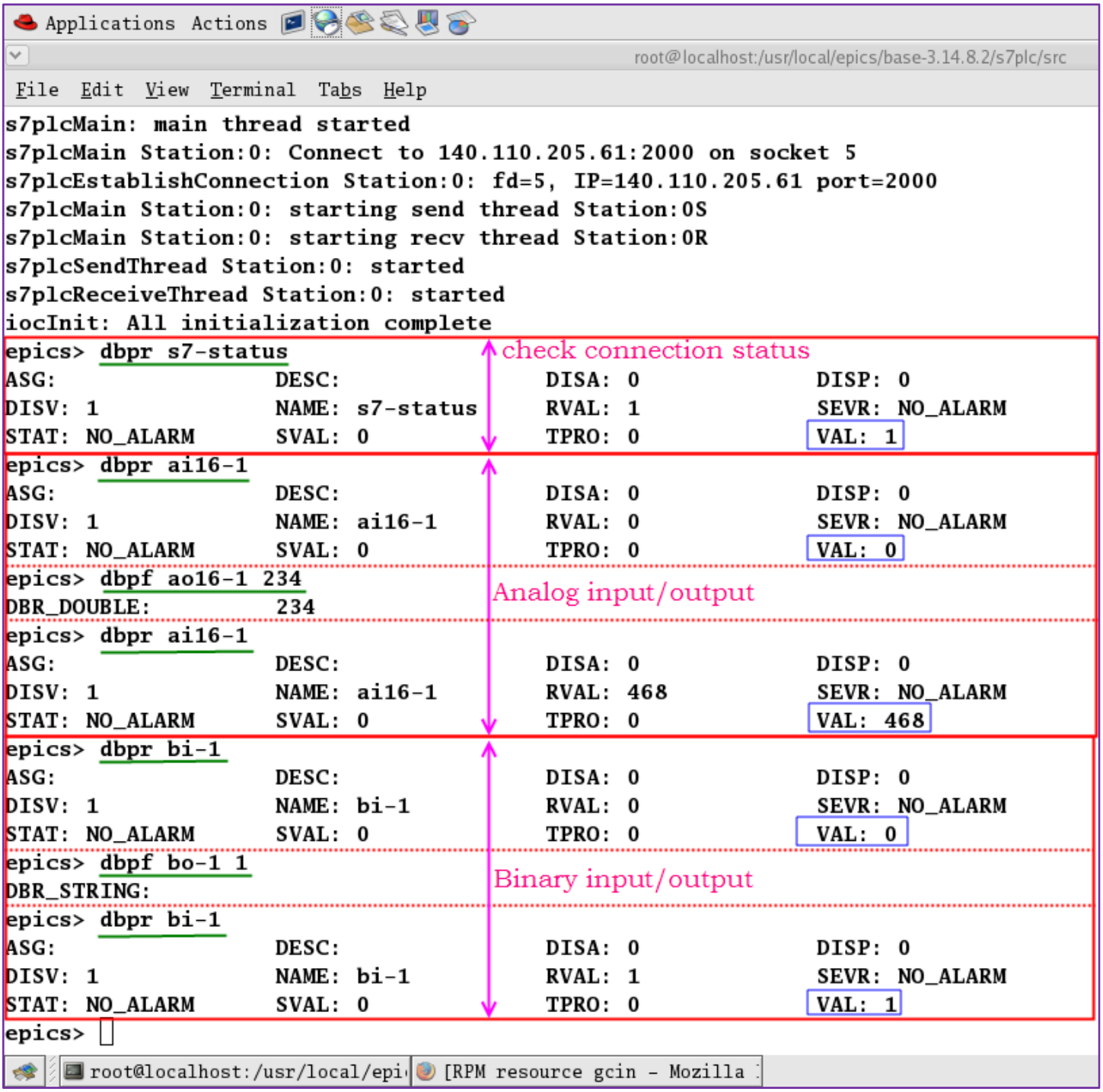


Figure 36

Execute the start up script. The result of the execution is illustrated in Figure 36. It is easy to read/write an analog value from/to the PLC and turn on/off a switch of a device.

Enjoy EPICS!