Siemens ™ S7 Protocol driver for OpenPLC

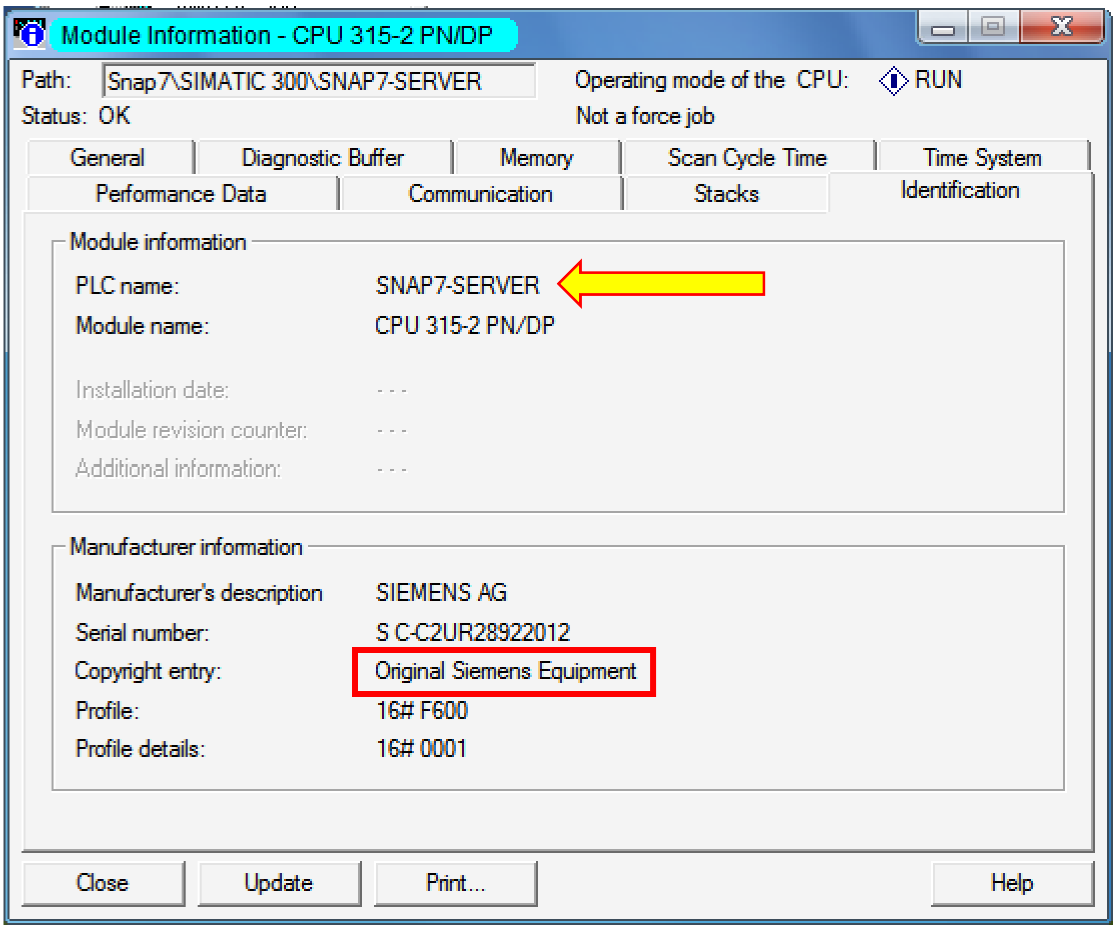
User Manual

Davide Nardella

Purpose of use

This driver allows **OpenPLC Runtime** to be "seen" from the outside as a Siemens PLC.

That is, any HMI/Scada that implements S7Protocol can connect, via ethernet to OpenPLC as if it were a **CPU 315-2PN/DP.**



Path: Snap7\SIMATIC 300\SNAP7-SERVER
Status: OK
Operating mode of the CPU: RUN
Not a force job

Module information

PLC name:	SNAP7-SERVER
Module name:	CPU 315-2 PN/DP
Installation date:	- - -
Module revision counter:	- - -
Additional information:	- - -

Manufacturer information

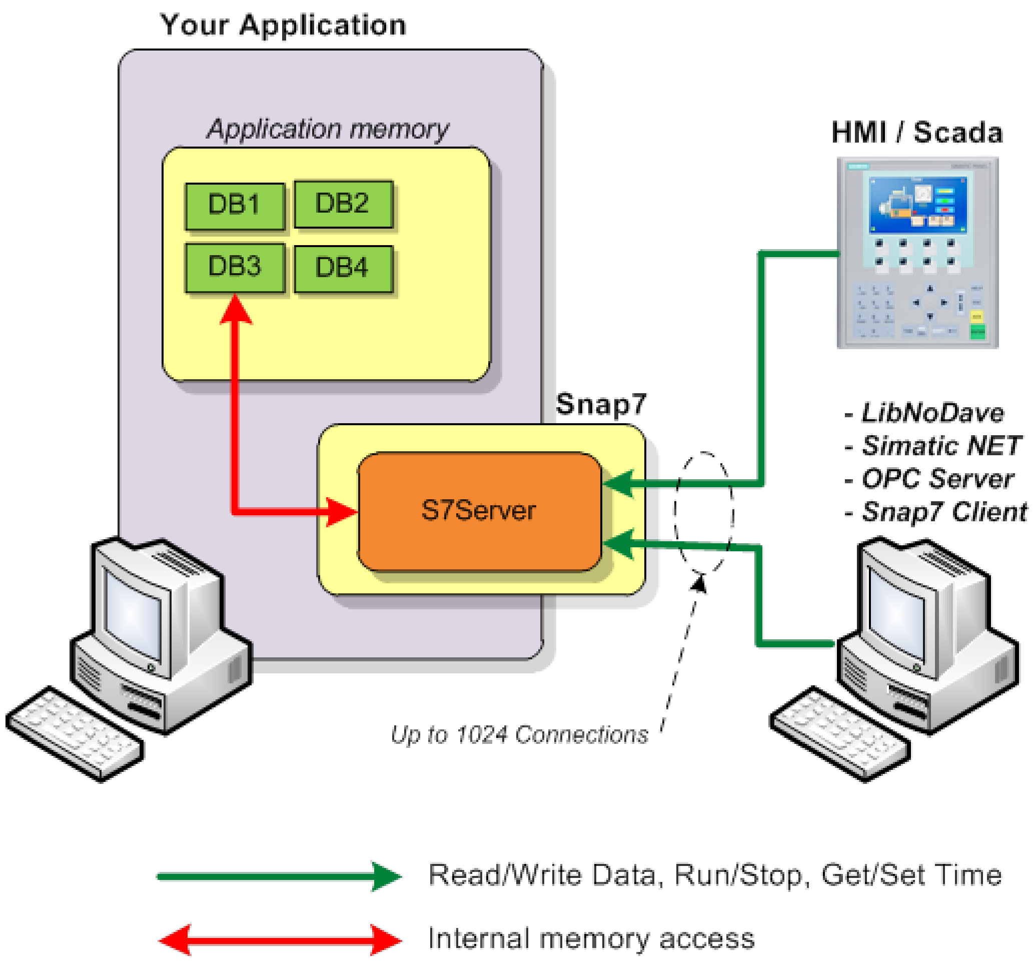
Manufacturer's description	SIEMENS AG
Serial number:	S C-C2UR28922012
Copyright entry:	Original Siemens Equipment
Profile:	16# F600
Profile details:	16# 0001

This is achieved using the **Snap7** library, specifically the **S7Server** object.

To be precise, S7Server does not fully simulate a PLC, but rather a CP (Communication Processor), which in PLCs can be contained in the CPU or be external such as a CP-343 unit.

Ultimately, this type of communication is similar to Modbus: an HMI/Scada or other equipment, connects to the PLC and reads/writes its internal variables.

S7Protocol is fast and secure, it is not only a protocol for data exchange, but is also used by Siemens for complete PLC programming. In fact, through S7Protocol, it is possible to communicate with a CPU even if it is completely empty or in STOP.



There are two versions of this protocol, the first, completely binary, capable of interfacing with both the 300/400 series CPUs and the second (the extended one) that works only on the TIA Portal CPUs (S712XX/S715XX). Snap7 implements the first version.

The Snap7 library also contains two other objects, **S7Client** and **S7Partner**. The first allows, as a client, to connect to a PLC, the second allows peer-to-peer communication with another Siemens PLC and can be an active or passive partner.

In this context only S7Server is used.

Technical features

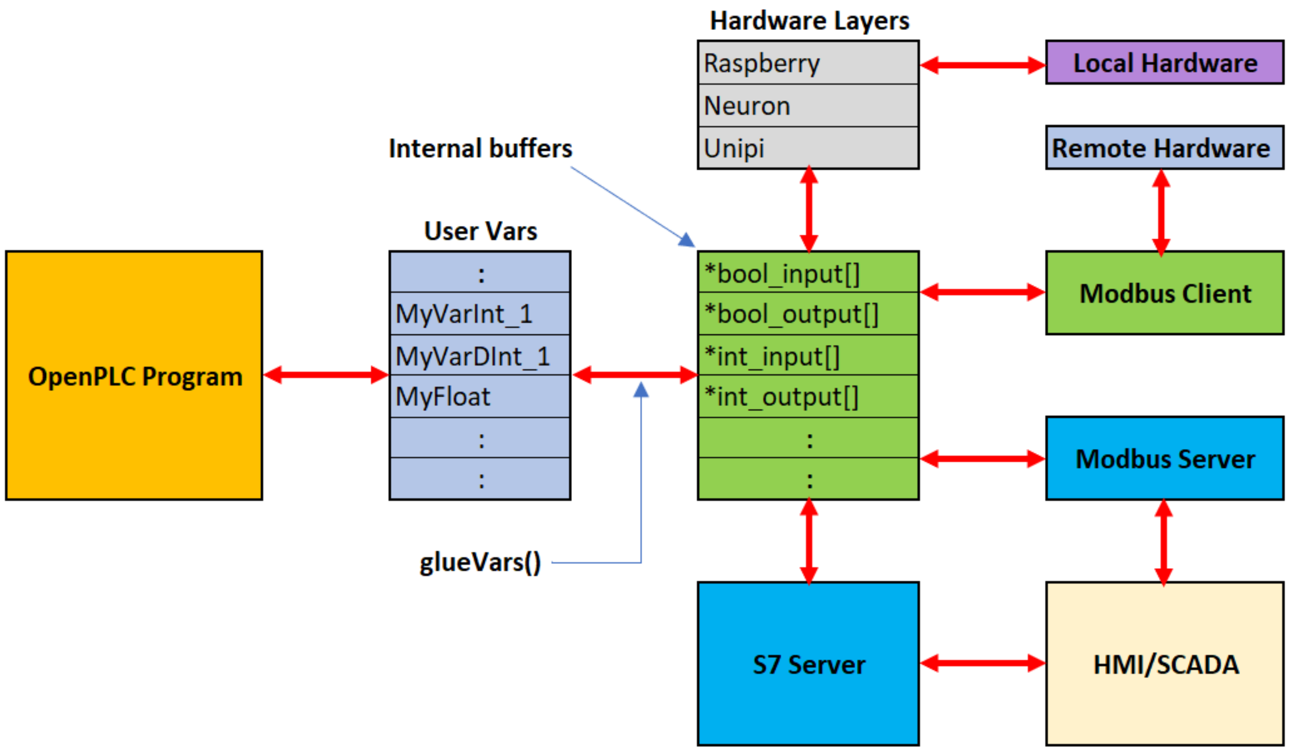
S7Server is a fully multithreaded TCP/IP server, it can handle up to 1024 simultaneous connections, each Server Socket works in an independent thread.

It is **zeroconf**, like a real Siemens communication processor. It has access to the internal buffers even if you have not declared variables associated with them.

Access to data is arbitrated with an intelligent lock: only the memory area to which access is granted is locked.

This driver does not depend on Modbus or other I/O drivers, it can work simultaneously with them.

Communication takes place, from the outside, directly with the internal areas of OpenPLC, as shown in the figure.



These features, combined with the intrinsic complexity of the protocol, justify the implementation in OpenPLC limited to the following platforms:

- **Windows**
- **Linux (X86/X64)**
- **Linux ARM (Raspberry and similar SBCs)**

Which have sufficient processing power and memory.

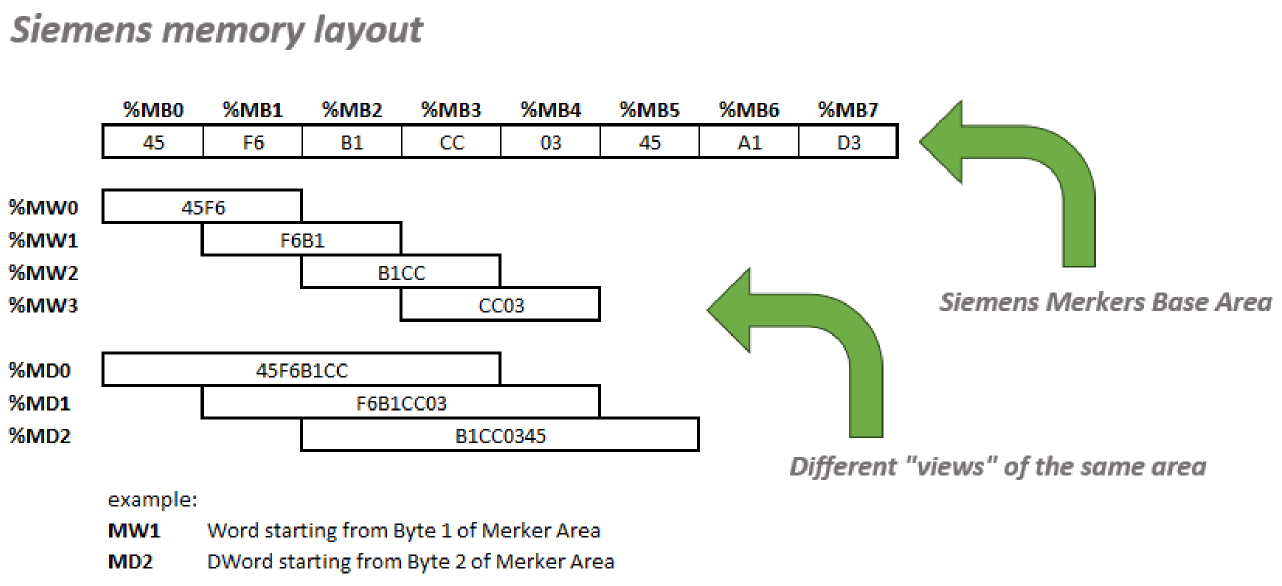
Data Layout

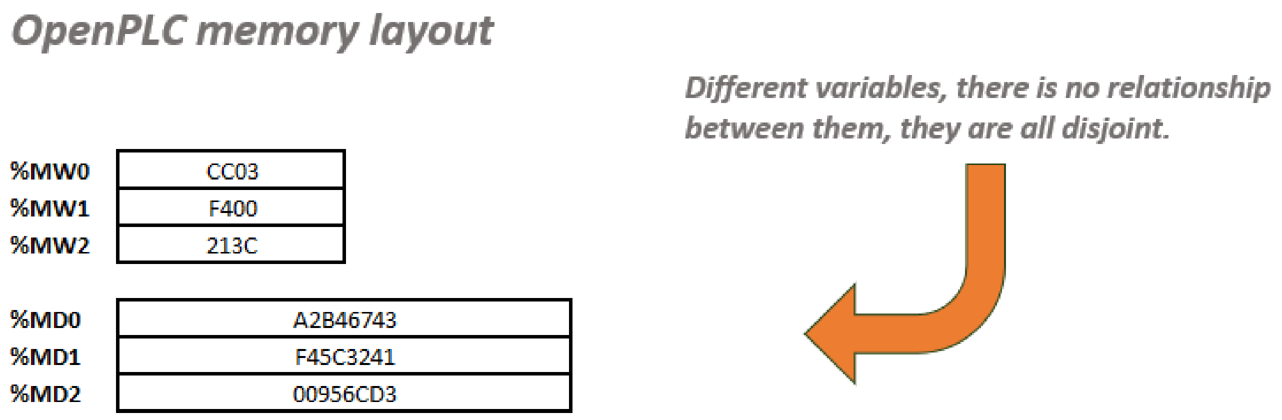
Although Siemens PLCs comply with the IEC61131-3 standard, the memory organization is very different from that of "native" IEC PLCs.

Let's see, specifically, the differences between a Siemens CPU and OpenPLC.

Siemens memory is a stream of bytes that can be accessed using "alias" addresses when we need to work with integers, double words or reals.

In OpenPLC, on the contrary, the variables are all separated from each other and grouped by size. It is important to understand this difference right away to avoid headaches later.

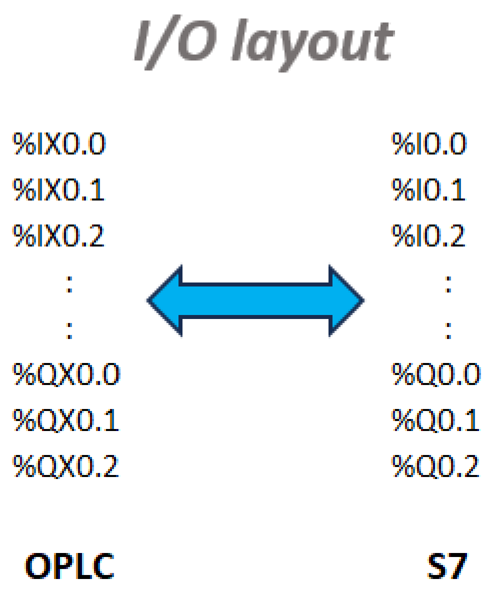

Siemens memory layout
%MB0 %MB1 %MB2 %MB3 %MB4 %MB5 %MB6 %MB7
45 F6 B1 CC 03 45 A1 D3
%MW0 45F6
%MW1 F6B1
%MW2 B1CC
%MW3 CC03
%MD0 45F6B1CC
%MD1 F6B1CC03
%MD2 B1CC0345
Siemens Merkers Base Area
Different "views" of the same area
example:
MW1 Word starting from Byte 1 of Merker Area
MD2 DWord starting from Byte 2 of Merker Area



In other words, a Siemens HMI expects %MW0 to be the high end of %MD0 (Siemens is big-endian) and %MW2 to be the low end.

While in OpenPLC there is no relationship between %MW0 and %MD0.

Digital Inputs/Outputs, fortunately, are mapped the same way.



Therefore, not being able to directly map variables (not I/O) with the same name, I used DB (data block) addressing which are converted into access to internal buffers, according to this table.

Resource type	PLC Address		Siemens Address	
	From	To	From	To
Digital Inputs	%IX0.0	%IX1023.7	%I0.0	%I1023.7
Digital Outputs	%QX0.0	%QX1023.7	%Q0.0	%Q1023.7
Analog Inputs	%IW0	%IW1023	%DB2.DBW0	%DB2.DBW2046
Analog Outputs	%QW0	%QW1023	%DB102.DBW0	%DB102.DBW2046
Int Memory	%MW0	%MW1023	%DB1002.DBW0	%DB1002.DBW2046
Dint Memory	%MD0	%MD1023	%DB1004.DBD0	%DB1004.DBD8188

For example, going to read/write the Siemens variable %DB2.DBW0, we will actually have access to %IW0.

Be careful, always use word-sized addresses in variables, otherwise the data will overlap across multiple variables.

Installation activation and configuration

Snap7 Server is installed during the installation of OpenPLC Runtime transparently.

Under Linux/Rpi the libraries are compiled on the fly to be aligned with the current **libc** version and the operating system architecture (32/64 bit).

Under Windows the binary DLLs compiled with Visual Studio 2022 are used, which are more compact and faster than those produced by mingw.

However, if you want, the complete distribution of Snap7 (which you can download from **Sourceforge**), also contains the makefiles to compile the libraries with mingw (32/64).

Activation

S7Server, by default, **is not active**. Activating it, however, is a very simple operation and is in line with what happens with other drivers for OpenPLC.

There is a text file named "s7protocol", you can find it in:

Windows:

C:\OpenPLC_Runtime\home\<user name>\OpenPLC_v3\webserver\scripts

Linux:

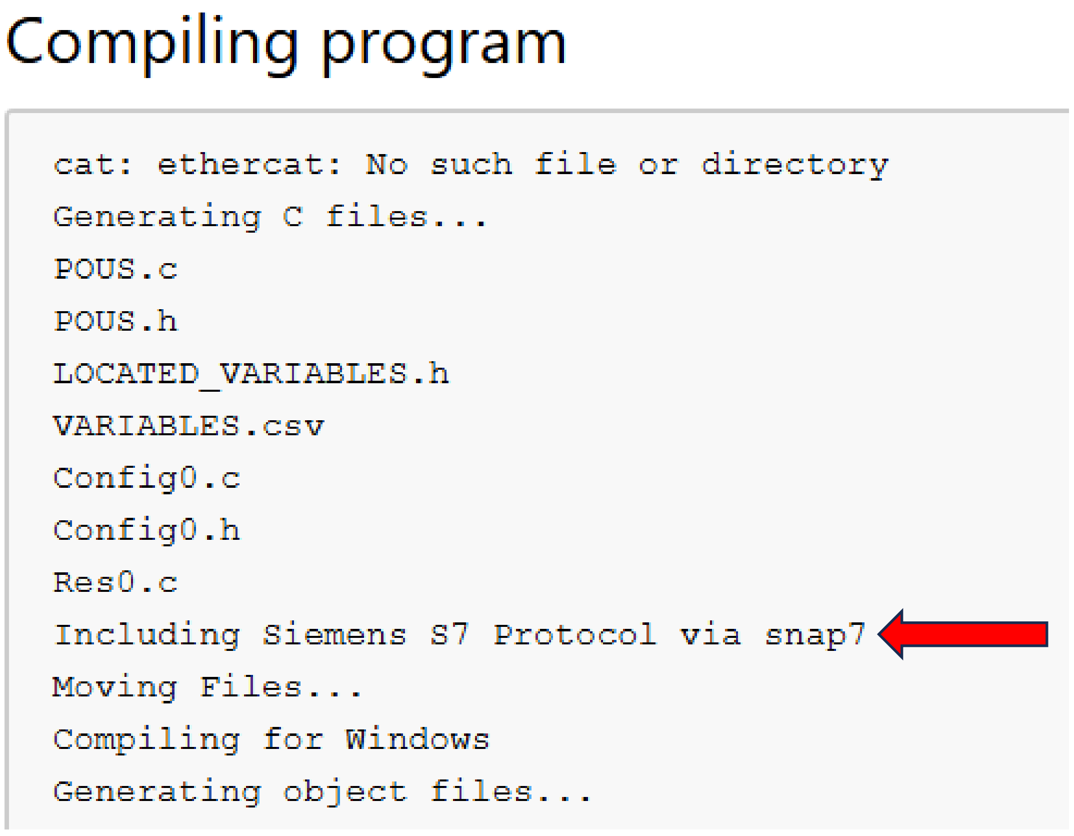
/home/<user name>/OpenPLC_v3/webserver/scripts

By default, it contains the string **#s7protocol**, to activate Snap7 simply delete the char **#** (don’t substitute it with a space).

You don't have to do anything else, as soon as you change or update the program in OpenPLC, the driver will be compiled with it.

To exclude Snap7, the procedure is reversed: insert the # character in the s7protocol file.

Excluding Snap7 will completely disable all references and will not include any files in the build.



Compiling program

```
cat: ethercat: No such file or directory
Generating C files...
POUS.c
POUS.h
LOCATED_VARIABLES.h
VARIABLES.csv
Config0.c
Config0.h
Res0.c
Including Siemens S7 Protocol via snap7
Moving Files...
Compiling for Windows
Generating object files...
```

Configuration

As said, there’s nothing to configure OpenPLC side.

in the next paragraph, as an example, we will see how to configure WinCC.

WinCC Unified

I did all the tests with **Siemens WinCC Unified V18.0**, because it is the newest and the most "demanding" environment, the connection must be perfect. In the past I noticed that with some compatible PLC it did not work. It is not the easiest software to use, but it is very structured.

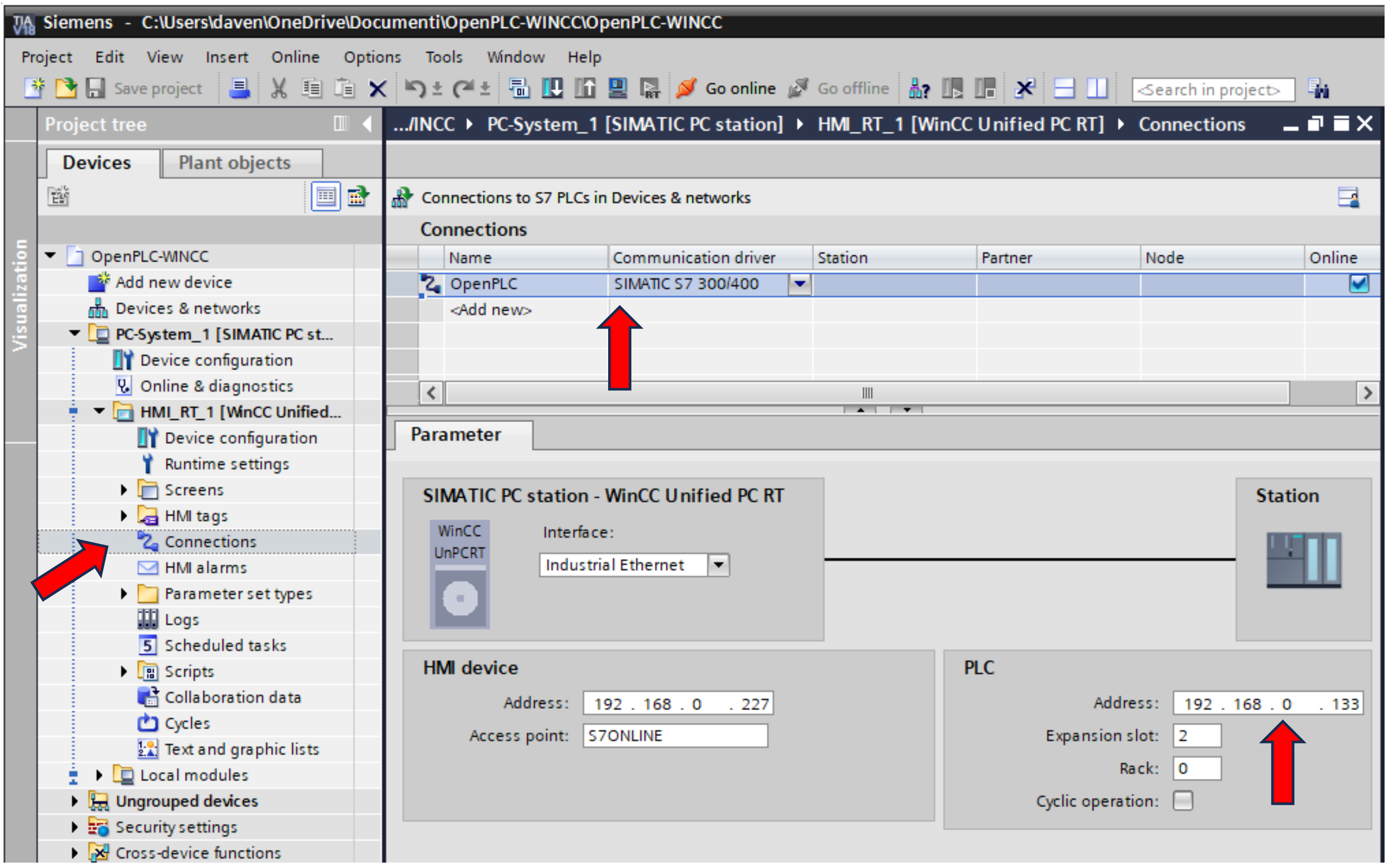
The process is quite standard:

- **Create a connection (which is the "channel" to the PLC with which we want to communicate)**
- **Create tags associated with that connection**
- **Create graphic pages**
- **Link graphic objects to the Tags above**

Connection

Create a new connection as in figure:

In Connections page double-click on **<Add new>** and select **SIMATIC S7300/400** as driver.



Connections

Name	Communication driver	Station	Partner	Node	Online
OpenPLC	SIMATIC S7 300/400				✓
<Add new>					

Interface: Industrial Ethernet

HMI device

Address: 192 . 168 . 0 . 227
Access point: S7ONLINE

PLC

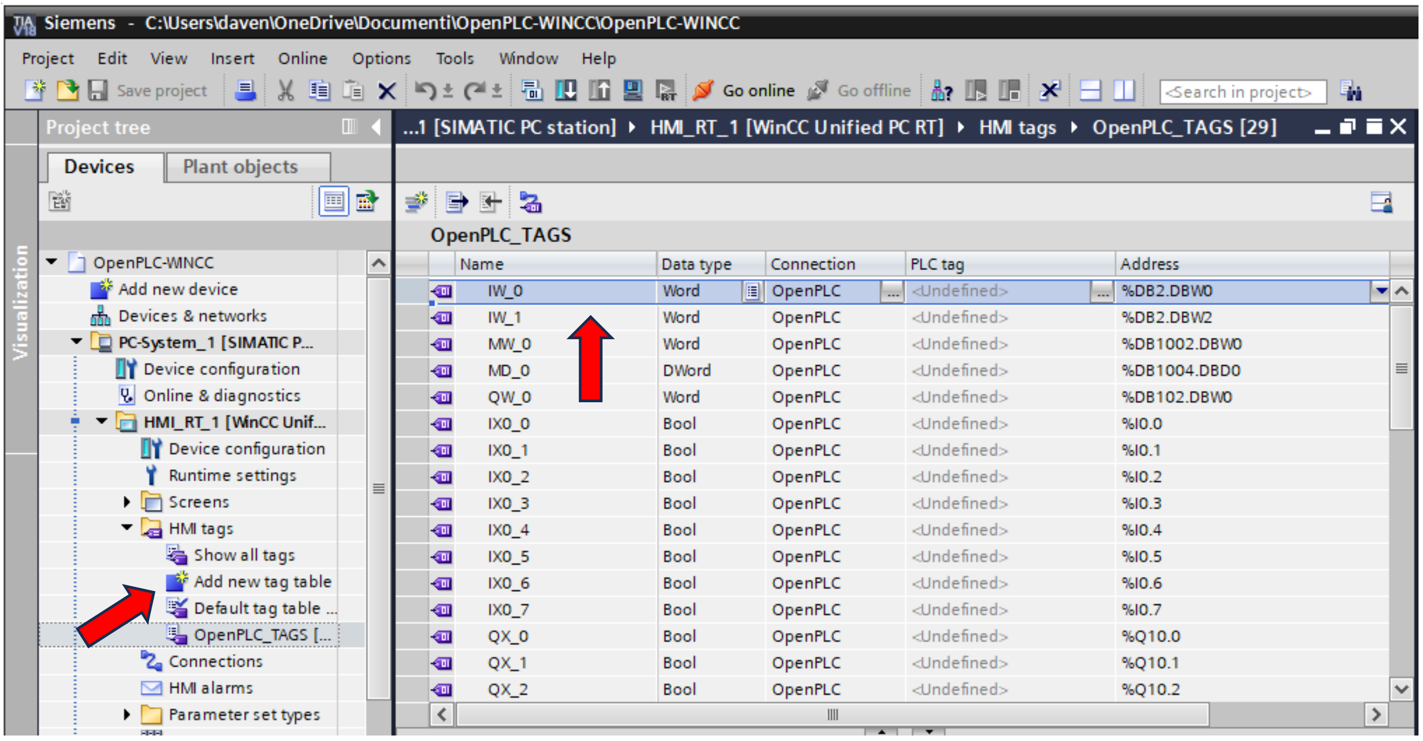
Address: 192 . 168 . 0 . 133
Expansion slot: 2
Rack: 0
Cyclic operation: ☐

Then insert the OpenPLC IP **Address.** Leave Rack=0 and Slot = 2.

Tags

Create a new group in **HMI tags**. Tags can be mixed across the connections, but I recommend you be neat, use a different group for each connection.

Then insert your tags.


Siemens - C:\Users\daven\OneDrive\Documenti\OpenPLC-WINCC\OpenPLC-WINCC

...1 [SIMATIC PC station] ▸ HMI_RT_1 [WinCC Unified PC RT] ▸ HMI tags ▸ OpenPLC_TAGS [29]

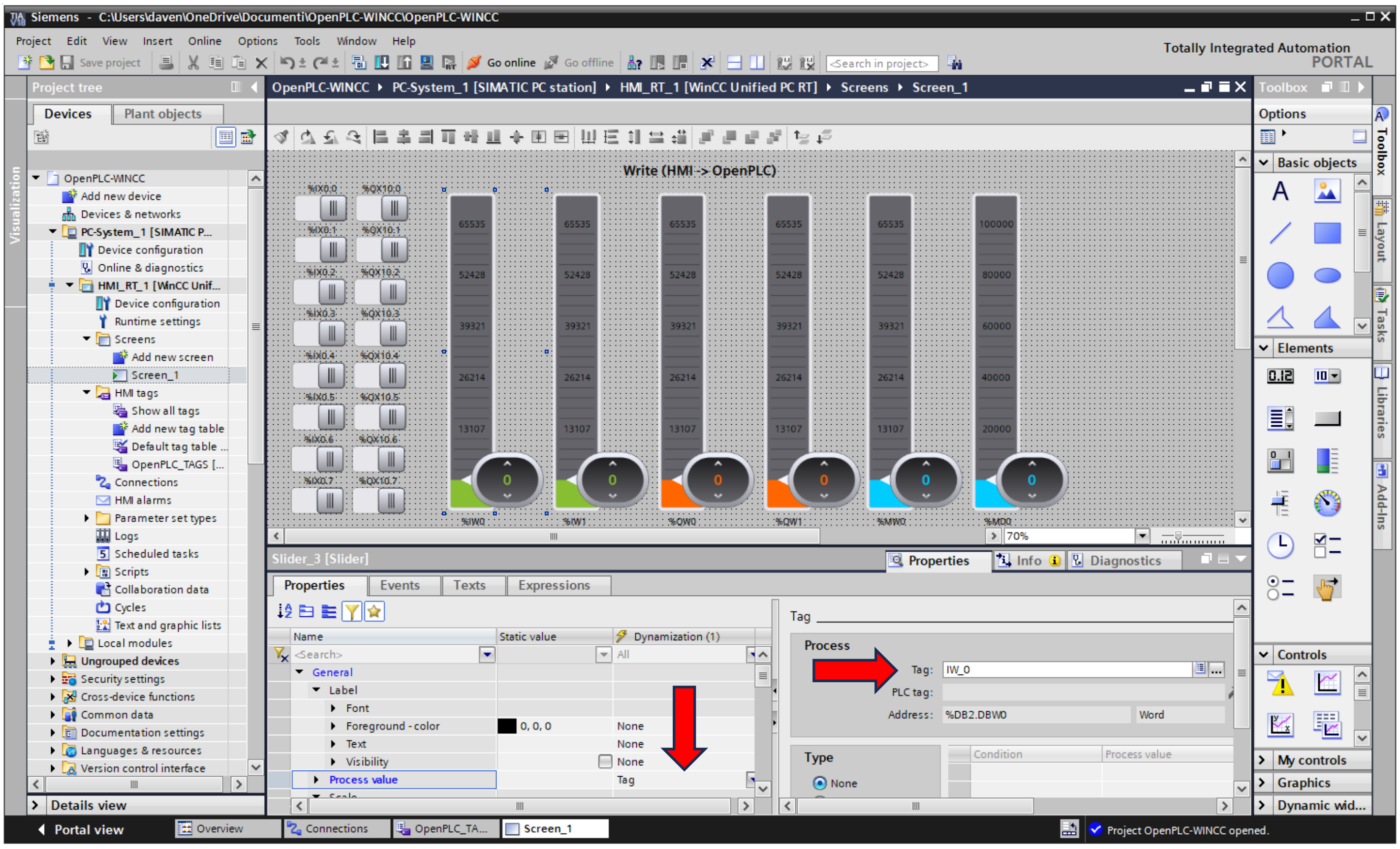
OpenPLC_TAGS

Name	Data type	Connection	PLC tag	Address
IW_0	Word	OpenPLC	<Undefined>	%DB2.DBW0
IW_1	Word	OpenPLC	<Undefined>	%DB2.DBW2
MW_0	Word	OpenPLC	<Undefined>	%DB1002.DBW0
MD_0	DWord	OpenPLC	<Undefined>	%DB1004.DBD0
QW_0	Word	OpenPLC	<Undefined>	%DB102.DBW0
IX0_0	Bool	OpenPLC	<Undefined>	%I0.0
IX0_1	Bool	OpenPLC	<Undefined>	%I0.1
IX0_2	Bool	OpenPLC	<Undefined>	%I0.2
IX0_3	Bool	OpenPLC	<Undefined>	%I0.3
IX0_4	Bool	OpenPLC	<Undefined>	%I0.4
IX0_5	Bool	OpenPLC	<Undefined>	%I0.5
IX0_6	Bool	OpenPLC	<Undefined>	%I0.6
IX0_7	Bool	OpenPLC	<Undefined>	%I0.7
QX_0	Bool	OpenPLC	<Undefined>	%Q10.0
QX_1	Bool	OpenPLC	<Undefined>	%Q10.1
QX_2	Bool	OpenPLC	<Undefined>	%Q10.2

Have a look at IW_0, we will use it.

In accord to our table DB2 is linked to %IW.

Graphics


Siemens - C:\Users\daven\OneDrive\Documenti\OpenPLC-WINCC\OpenPLC-WINCC

OpenPLC-WINCC ▸ PC-System_1 [SIMATIC PC station] ▸ HMI_RT_1 [WinCC Unified PC RT] ▸ Screens ▸ Screen_1

Write (HMI -> OpenPLC)

Slider_3 [Slider]

Name	Static value	Dynamization (1)
General		
Label		
Font		
Foreground - color	0, 0, 0	None
Text		None
Visibility		None
Process value		Tag

Tag

Process

Tag: IW_0

PLC tag:

Address: %DB2.DBW0 Word

Type

None

Project OpenPLC-WINCC opened.

Drag a slider from the toolbox, select it and press **Alt-Enter** to edit its properties.

Change the range the color and whatever you want, then in the field **Process Value** in the column **Dynamization** chose the option **Tag**, a right panel will appear.

Chose IW_0 using the symbols browser that will open pressing the [...] button.

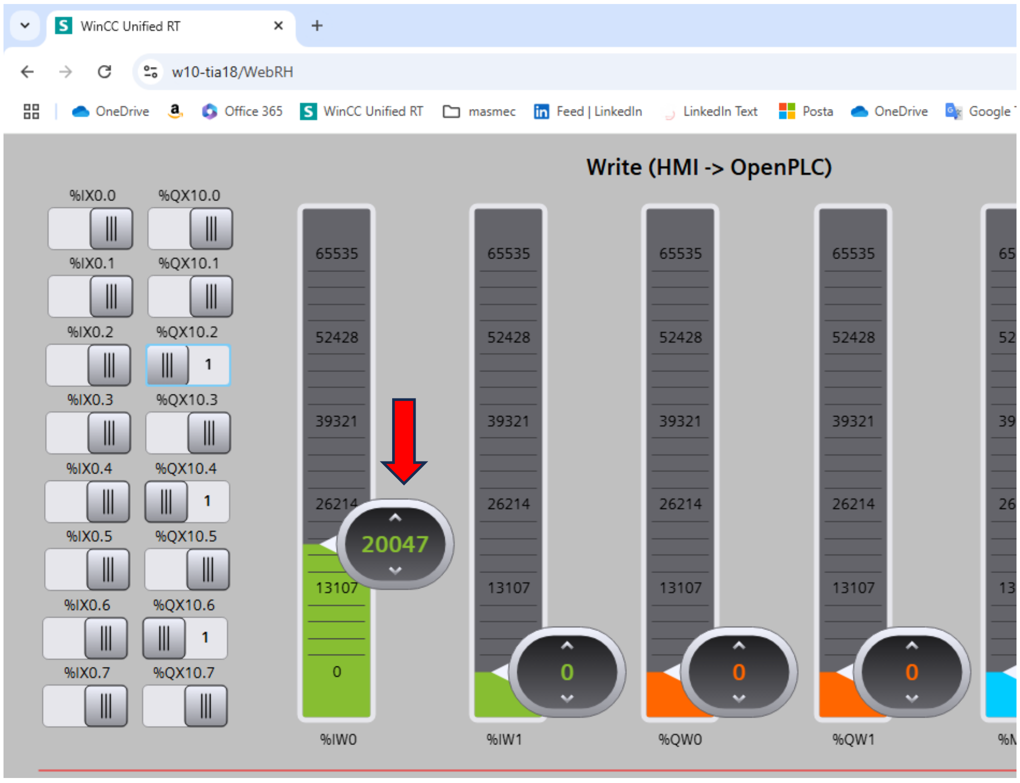
Compile and download.

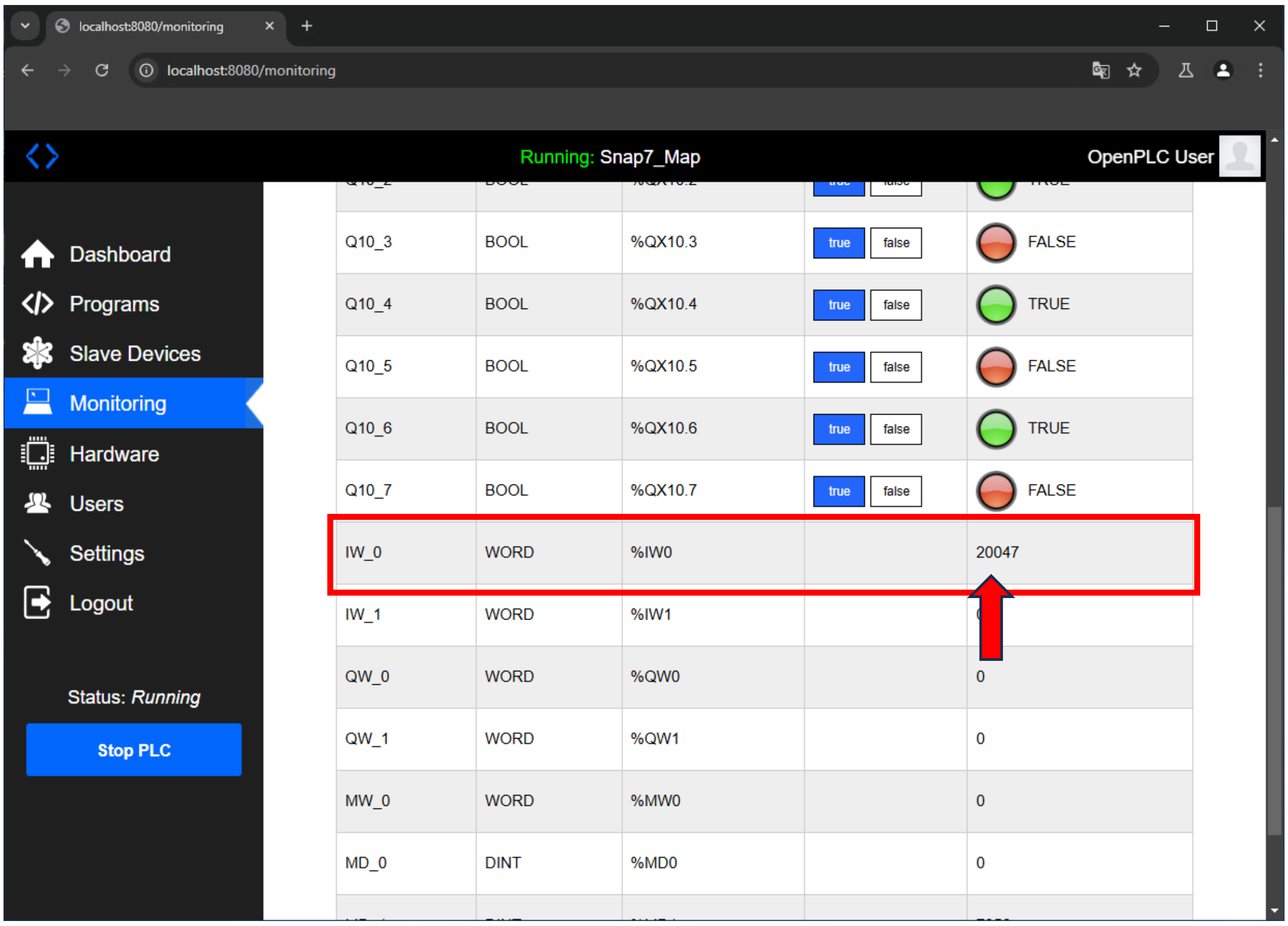
I focused only on the OpenPLC interface and omitted all the configurations to be done in the virtual rack, in the WinCC web server and in the S7ONLINE interface of PG/AG. (only the "getting started manual" is 140 pages).

But if you are a Siemens programmer you are already used to these superhuman sufferings. ☺

Now it's time to start OpenPLC Runtime…

Everything runs





Q10_3	BOOL	%QX10.3	true false	🟥 FALSE
Q10_4	BOOL	%QX10.4	true false	🟩 TRUE
Q10_5	BOOL	%QX10.5	true false	🟥 FALSE
Q10_6	BOOL	%QX10.6	true false	🟩 TRUE
Q10_7	BOOL	%QX10.7	true false	🟥 FALSE
IW_0	WORD	%IW0		20047
IW_1	WORD	%IW1		
QW_0	WORD	%QW0		0
QW_1	WORD	%QW1		0
MW_0	WORD	%MW0		0
MD_0	DINT	%MD0		0