PLC Python Siemens | Medium

Simone Belometti ⁝⁝ 2021/9/8

How to communicate with a Siemens PLC using Python

This is my first article. So, i apologize in advance for any errors. Any kind of suggestions are welcomed.

I work in manifactury company.
In the last year the management has decided to buy several machines.
To improve productivity and integration with the production process, the information system must be able to communicate with the machines.
For example, it is interesting to send the setting directly to the machine for the next production, or to monitor the execution of the work from the office.

Many machines consist of a PLC, often Siemens S7.
For the machines I had to analyze, the SIemens PLC was a 1200 model.

For various reasons I decided to use Python as a programming language.
I love Python for one simple reason. Think of a problem, in Python you can always find the library that's right for you.

Basically my application uses Django and Postgres, but in this article I would like to describe the use of Python to communicate with the PLC with the python-snap7 library.

Now I will describe a simple example of communication between PC and PLC via Python with python-snap7 library.
Let's assume that Python and pip are already installed on your PC.

Let's install the library.

```
pip install python-snap7
```

Good! We are ready.

For example, the PLC in the machine has an IP address: 192.168.50.50.

Let's write a simple Python program to read and write some variables inside the PLC.

Import the library

```
>>> import snap7
>>> from snap7 import util
```

I create the client instance for the connection to the PLC

```
>>> client = snap7.client.Client()
>>> client.connect('192.168.50.50',0,0)
```

I can check if i am actually connected to the PLC

```
>>> client.get_connected()
True
```

If you have come this far, you are right.

Now you have to figure out which db (memory area) you can read or write in the PLC.
For example, let's read database number 5, from byte 0 to byte 18

```
>>> client.db_read(5, 0, 18)
bytearray(b'\xc4\x9d\xc0\x00\xc2\xfcff\xff~\xc3\x02\x00\x00\xc3\x01\x00\x00')
```

Perfect. Now let's read for example a variable of type real.

```
>>> db = client.db_read(5, 0, 4)
>>> t = util.get_real(db, 0)
>>> print(t)
-1102.0
```

For writing the steps are the same.

```
>>> db = client.db_read(5, 0, 4)
>>> util.set_real(db, 0, 2.1)
```

Well done!
Simple right?

Now enjoy. I leave links to the libraries