

Electroválvula SMC EX260-SEN1

Una vez establecida la IP, hacemos ping para comprobar su conexión.

```
C:\Program Files (x86)\VMware\VMware Workstation\bin>ping 192.168.0.202

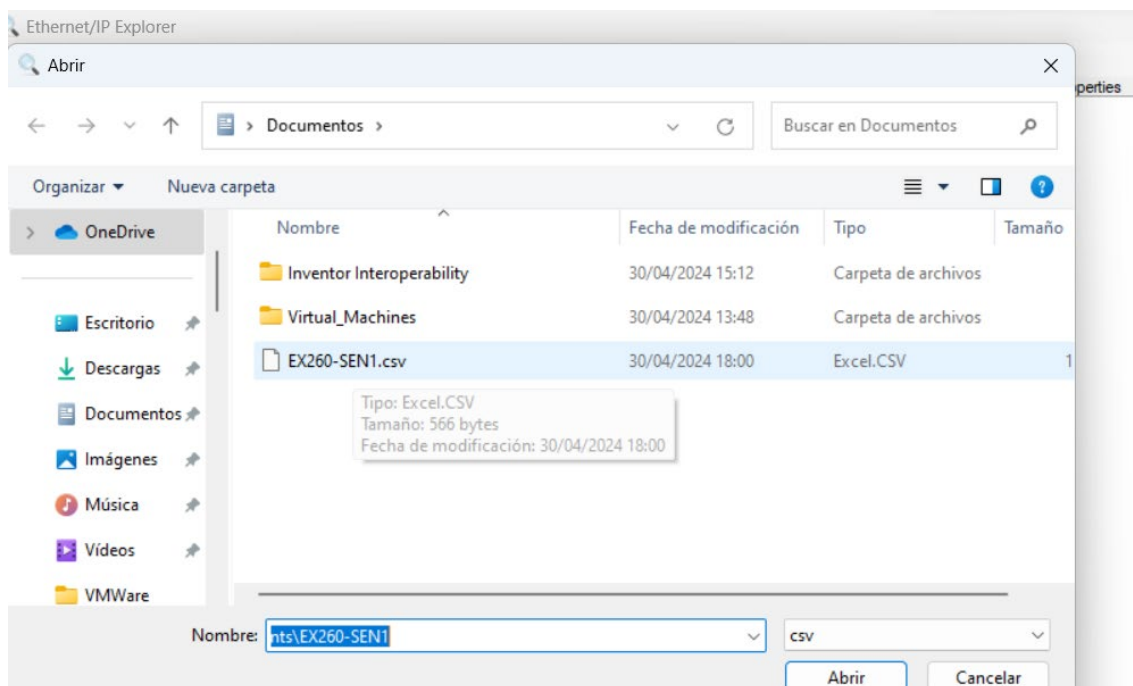
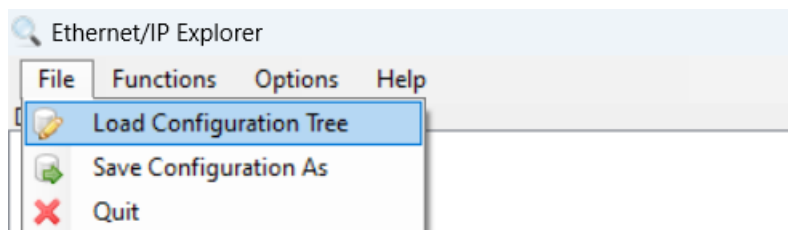
Haciendo ping a 192.168.0.202 con 32 bytes de datos:
Respuesta desde 192.168.0.202: bytes=32 tiempo<1m TTL=60
Respuesta desde 192.168.0.202: bytes=32 tiempo=1ms TTL=60
Respuesta desde 192.168.0.202: bytes=32 tiempo=1ms TTL=60
Respuesta desde 192.168.0.202: bytes=32 tiempo=1ms TTL=60

Estadísticas de ping para 192.168.0.202:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
        (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 0ms, Máximo = 1ms, Media = 0ms
```

Ejecutamos EnIPEXplorer:

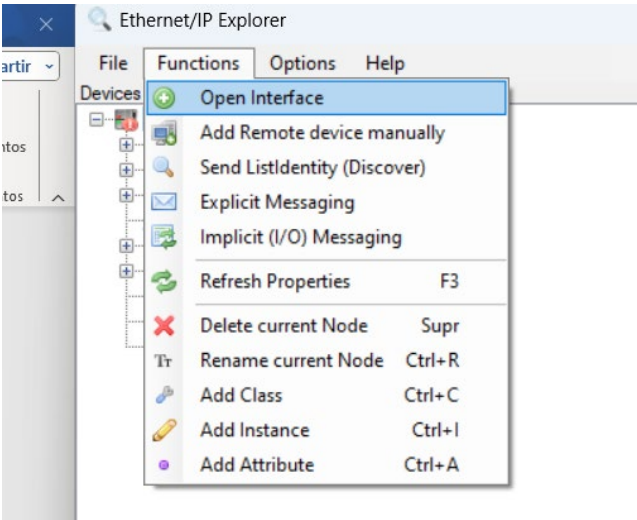


Cargamos el archivo correspondiente, pero también podemos crearlo de cero:

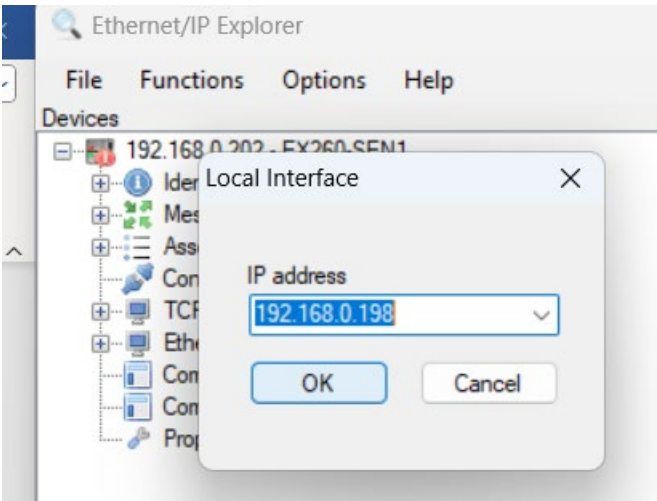


Electroválvula SMC EX260-SEN1

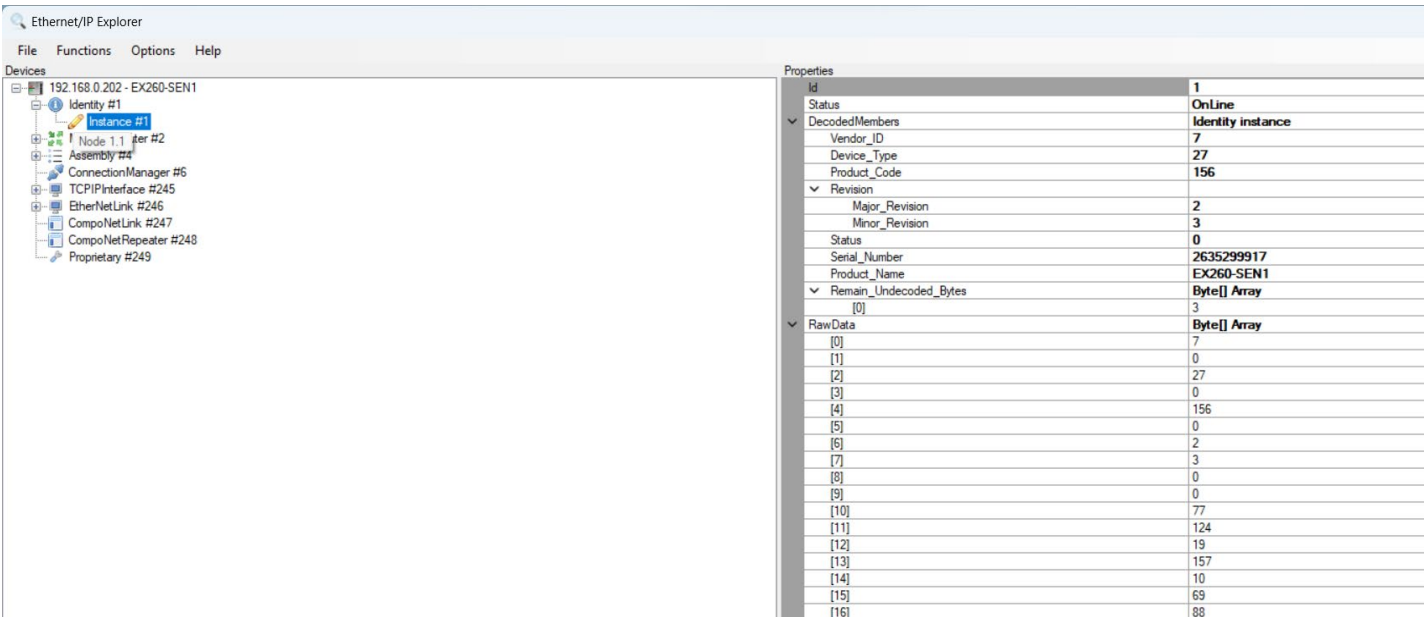
Una vez cargado, abrimos la conexión:



Nos pedirá la IP de nuestro Host:



Una vez abierta la conexión, podemos ver la identidad y los atributos, que también están representados en el archivo de configuración EDS.



Electroválvula SMC EX260-SEN1

Para escribir necesitamos ir al área de salida, para ello debemos ir a la Assembly #4, Interface #150 y Attribute #3, para escribir debemos utilizar la primera posición del Byte[] Array que esta asignada al Byte[0].

Ethernet/IP Explorer

File Functions Options Help

Devices

192.168.0.202 - EX260-SEN1

Identity #1

Instance #1

MessageRouter #2

Assembly #4

Instance #1

Instance #100

Instance #150

Attribute #3

ConnectionManager #4

Node 4.150.3

TCP/IPInterface #245

EtherNetLink #246

CompoNetLink #247

CompoNetRepeater #248

Proprietary #249

Properties

Id	3
Status	OnLine
DecodedMembers	
RawData	Byte[] Array
[0]	0
[1]	0
[2]	0
[3]	0

Ejemplos de escritura:

Ethernet/IP Explorer

File Functions Options Help

Devices

192.168.0.202 - EX260-SEN1

Identity #1

Instance #1

MessageRouter #2

Assembly #4

Instance #1

Instance #100

Instance #150

Attribute #3

ConnectionManager #6

TCP/IPInterface #245

EtherNetLink #246

CompoNetLink #247

CompoNetRepeater #248

Proprietary #249

Properties

Id	3
Status	OnLine
DecodedMembers	
RawData	Byte[] Array
[0]	1
[1]	0
[2]	0
[3]	0

Log

GetAttributesAll : Service_not_supported - Node 4.150 - Endpoint 192.168.0.202:44818

Write OK

Write OK

Write OK

Write OK

Write OK

Write OK

Ethernet/IP Explorer

File Functions Options Help

Devices

192.168.0.202 - EX260-SEN1

Identity #1

Instance #1

MessageRouter #2

Assembly #4

Instance #1

Instance #100

Instance #150

Attribute #3

ConnectionManager #6

TCP/IPInterface #245

EtherNetLink #246

CompoNetLink #247

CompoNetRepeater #248

Proprietary #249

Properties

Id	3
Status	OnLine
DecodedMembers	
RawData	Byte[] Array
[0]	12
[1]	0
[2]	0
[3]	0

Electroválvula SMC EX260-SEN1

Ethernet/IP Explorer

File Functions Options Help

Devices

192.168.0.202 - EX260-SEN1

Identity #1

Instance #1

MessageRouter #2

Assembly #4

Instance #1

Instance #100

Instance #150

Attribute #3

ConnectionManager #6

TCP/IPInterface #245

EtherNetLink #246

CompoNetLink #247

CompoNetRepeater #248

Proprietary #249

Properties

Id

Status

DecodedMembers

RawData

[0]

[1]

[2]

[3]

3

Online

Byte[] Array

24

0

0

0

Para escribir en el estado de todas, utilizamos el entero 127.

Ethernet/IP Explorer

File Functions Options Help

Devices

192.168.0.202 - EX260-SEN1

Identity #1

Instance #1

MessageRouter #2

Assembly #4

Instance #1

Instance #100

Instance #150

Attribute #3

ConnectionManager #6

TCP/IPInterface #245

EtherNetLink #246

CompoNetLink #247

CompoNetRepeater #248

Proprietary #249

Properties

Id

Status

DecodedMembers

RawData

[0]

[1]

[2]

[3]

3

Online

Byte[] Array

127

0

0

0

Electroválvula SMC EX260-SEN1

Para poder manipularlas hay varias librerías en diferentes lenguajes, disponibles en GitHub, ejemplo de la librería cpppo de Python:

En el nos conectamos a la electroválvula y especificamos en @4/150/3, el Assembly/Instance y Attribute.

```
import cpppo.py > ...
1  from cpppo.server.enip import client
2  from cpppo.server.enip.get_attribute import attribute_operations
3  from cpppo.server.enip.get_attribute import proxy_simple
4
5
6  # Dirección IP del dispositivo EtherNet/IP
7
8  HOST = "192.168.0.202"
9
10 # Dirección del área de salida
11 output_area_address = ("@4/150/3")
12
13
14 # Leer el valor en el área de entrada usando el cliente
15 TAGS = [output_area_address]
16
17
18 # Crear un objeto proxy simple
19 via = proxy_simple(HOST)
20
21
22 with client.connector(host=HOST) as conn:
23     for index, descr, op, reply, status, value in conn.synchronous(
24         operations=attribute_operations(
25             TAGS, route_path=[], send_path='' ):
26
27         print("Índice: %s" % index)
28         print("Operación: %s" % op)
29         print("Respuesta: %s" % reply)
30         print("Estado: %s" % status)
31         print(": %s" % ( descr))
32         print(value)
```

Salida del programa con las electroválvulas en 0:

```
PS C:\repo> & C:/Python34/python.exe "c:/repo/import cpppo.py"
Índice: 0

Operación: {'service': 14, 'input': bytearray(b'\x0e\x03 \x04$\x960\x03'), 'get_attribute_single': True, 'path': {'segment': [{'class': 4}, {'instance': 150}, {'attribute': 3}]}}

Respuesta: {'input': array('B', [142, 0, 0, 0, 0, 0, 0, 0]), 'service': 142, 'status': 0, 'get_attribute_single': {'data': [0, 0, 0, 0]}, 'status_ext': {'size': 0}}

Estado: 0

: Single G_A_S      @0x0004/150/3

[0, 0, 0, 0]
```

Electroválvula SMC EX260-SEN1

Salida del programa con las electroválvulas en 127:

```
import cpppo.py > ...
1 from cpppo.server.enip import client
2 from cpppo.server.enip.get_attribute import attribute_operations
3 from cpppo.server.enip.get_attribute import proxy_simple
4
5
6 # Dirección IP del dispositivo EtherNet/IP
7
8 HOST = "192.168.0.202"
9
10 # Dirección del área de salida
11 output_area_address = ("@4/150/3")
12
13
14 # Leer el valor en el área de entrada usando el cliente
15 TAGS = [output_area_address]
16
17
```

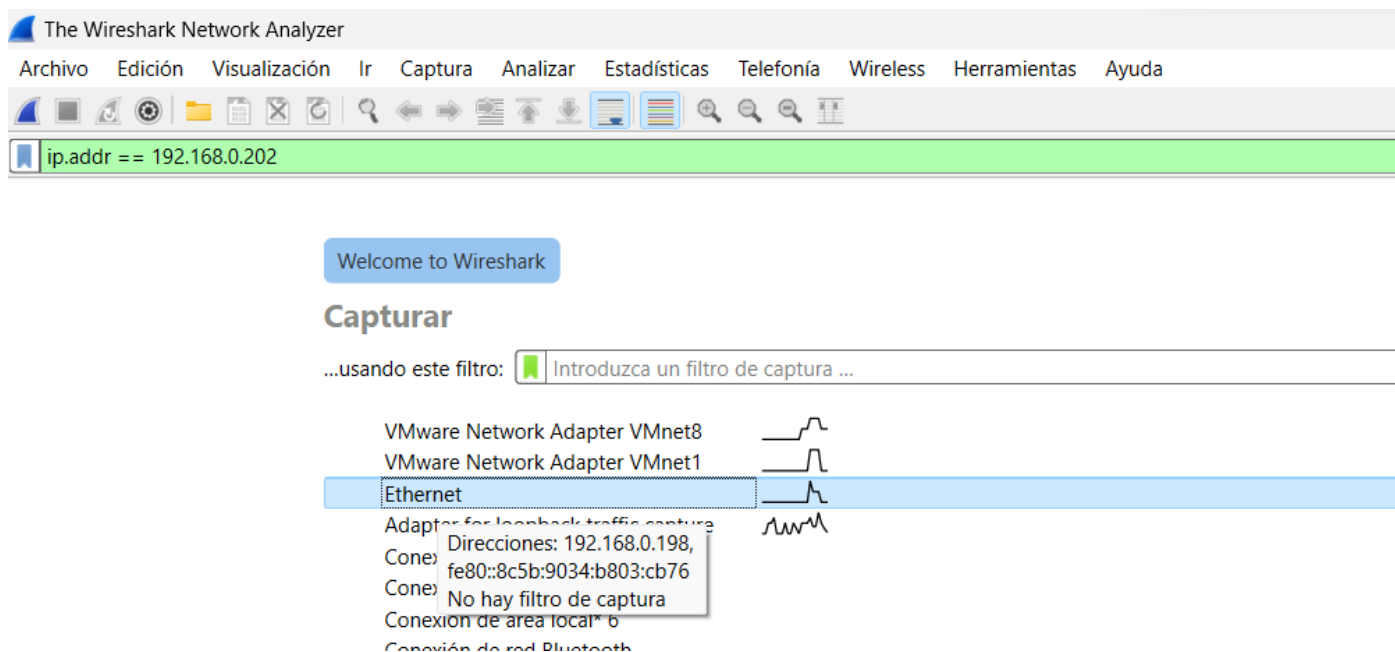
PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS

PS C:\repo> & C:/Python34/python.exe "c:/repo/import cpppo.py"

Índice: 0
Operación: {'get_attribute_single': True, 'path': {'segment': [{'class': 4}, {'instance': 150}, {'attribute': 3}]}, 'input': bytearray(b'\x0e\x03 \x04\x960\x03'), 'service': 14}
Respuesta: {'get_attribute_single': {'data': [127, 0, 0, 0]}, 'input': array('B', [142, 0, 0, 0, 127, 0, 0, 0]), 'status': 0, 'service': 142, 'status_ext': {'size': 0}}
Estado: 0
: Single G_A_S @0x0004/150/3
[127, 0, 0, 0]
PS C:\repo>

Capturar tramas para utilizar en diferentes lenguajes con Wireshark

Colocamos de filtro el ip.addr == IP de la electroválvula.



Electroválvula SMC EX260-SEN1

En esta primera trama podemos capturar cuando las valvulas se activan todas a la vez, en el Set Attribute Single, una vez activada nos devolverá el estado actualizado de la valvula mediante Get.

En la línea 0060 podemos ver la ruta especificada en el archivo EDS que es 20 04 24 96 30 03 y la activación que corresponde a 7F 00 00 00

Capturando desde Ethernet

ArchivoEdiciónVisualizaciónIrCapturaAnalizarEstadísticasTelefoníaWirelessHerramientasAyuda

ip.addr == 192.168.0.202

Info	No.	Time	Source	Destination	Protocol	Length
Assembly - Set Attribute Single	1	0.000000	192.168.0.198	192.168.0.202	CIP	
Success: Assembly - Set Attribute Single	2	0.001709	192.168.0.202	192.168.0.198	CIP	
Assembly - Get Attribute Single	3	0.002052	192.168.0.198	192.168.0.202	CIP	
Success: Assembly - Get Attribute Single	4	0.002469	192.168.0.202	192.168.0.198	CIP	
50778 → 44818 [ACK] Seq=101 Ack=93 Win=63344 Len=0	5	0.042701	192.168.0.198	192.168.0.202	TCP	

> Frame 1: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface \Device\NPF_{E13F8793-0000-0023-c6-7c} 13 9d 74 d4 dd 28 e4 aa 08 00 45 00 00 5c 58 22 40 00 00 06 00 00 c0 a8 00 c6 c0 a8 00 ca c6 5a af 12 9c 2f c3 18 04 16 0a 9d 50 18 f7 cc 83 2f 00 00 6f 00 1c 00 06 00 4d 7c 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 00 00 00 00 b2 00 00 0c 00 10 03 20 04 24 96 30 03 7f 00 00 00

> Ethernet II, Src: QuantaComput_28:e4:aa (74:d4:dd:28:e4:aa), Dst: SMC_7c:13:9d (00:23:c6:7c:13:9d)

> Internet Protocol Version 4, Src: 192.168.0.198, Dst: 192.168.0.202

> Transmission Control Protocol, Src Port: 50778, Dst Port: 44818, Seq: 1, Ack: 1, Len: 52

> EtherNet/IP (Industrial Protocol), Session: 0x7C4D0006, Send RR Data

> Common Industrial Protocol

Ahora hacemos lo mismo pero desactivando todas las válvulas, entonces vemos en la línea 0060 la modificación en 20 04 24 96 30 03 y la activación que corresponde a 00 00 00 00

ip.addr == 192.168.0.202

Info	No.	Time	Source	Destination	Protocol	Length
Assembly - Set Attribute Single	1	0.000000	192.168.0.198	192.168.0.202	CIP	10
Success: Assembly - Set Attribute Single	2	0.001709	192.168.0.202	192.168.0.198	CIP	9
Assembly - Get Attribute Single	3	0.002052	192.168.0.198	192.168.0.202	CIP	10
Success: Assembly - Get Attribute Single	4	0.002469	192.168.0.202	192.168.0.198	CIP	10
50778 → 44818 [ACK] Seq=101 Ack=93 Win=63344 Len=0	5	0.042701	192.168.0.198	192.168.0.202	TCP	5
Assembly - Set Attribute Single	8	15.961036	192.168.0.198	192.168.0.202	CIP	10
Success: Assembly - Set Attribute Single	9	15.962613	192.168.0.202	192.168.0.198	CIP	9
Assembly - Get Attribute Single	10	15.962765	192.168.0.198	192.168.0.202	CIP	10
Success: Assembly - Get Attribute Single	11	15.963592	192.168.0.202	192.168.0.198	CIP	10
50778 → 44818 [ACK] Seq=201 Ack=185 Win=63252 Len=0	12	16.008609	192.168.0.198	192.168.0.202	TCP	5

> Frame 8: 106 bytes on wire (848 bits), 106 bytes captured (848 bits) on interface \Device\NPF_{E13F8793-0000-0023-c6-7c} 13 9d 74 d4 dd 28 e4 aa 08 00 45 00 00 5c 58 25 40 00 00 06 00 00 c0 a8 00 c6 c0 a8 00 ca c6 5a af 12 9c 2f c3 7c 04 16 0a f9 50 18 f7 70 83 2f 00 00 6f 00 1c 00 06 00 4d 7c 00 00 00 00 00 00 00 00 00 00 00 00 02 00 00 00 00 b2 00 00 0c 00 10 03 20 04 24 96 30 03 00 00 00 00

> Ethernet II, Src: QuantaComput_28:e4:aa (74:d4:dd:28:e4:aa), Dst: SMC_7c:13:9d (00:23:c6:7c:13:9d)

> Internet Protocol Version 4, Src: 192.168.0.198, Dst: 192.168.0.202

> Transmission Control Protocol, Src Port: 50778, Dst Port: 44818, Seq: 101, Ack: 93, Len: 52

> EtherNet/IP (Industrial Protocol), Session: 0x7C4D0006, Send RR Data

> Common Industrial Protocol