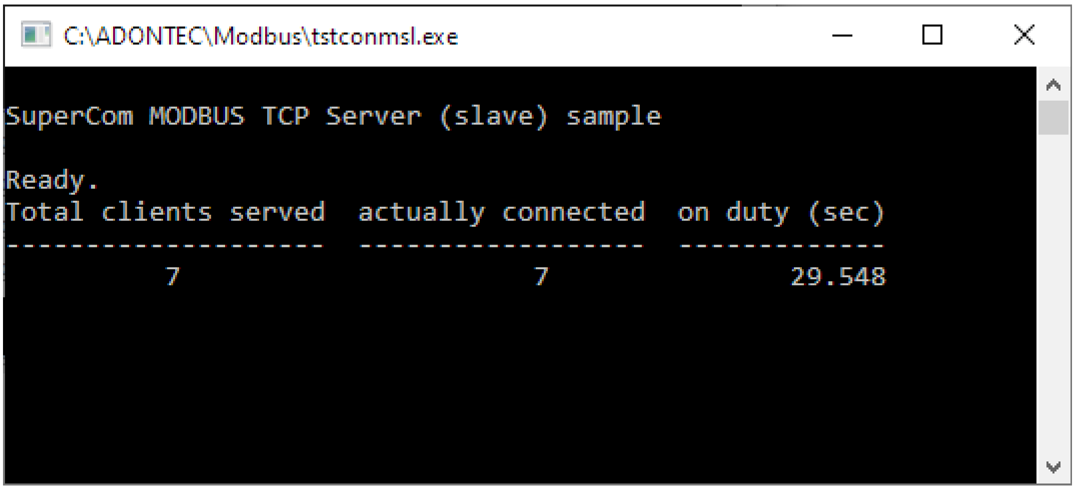
SuperCom MODBUS Protocol Module

The following are some of the server (slave) and client examples that are included with the product. Many more are on the product file.

Since a server (slave) usually supports more than one client simultaneously some of the samples show the „actually connected“, which is the number of clients currently served simultaneously by the slave.

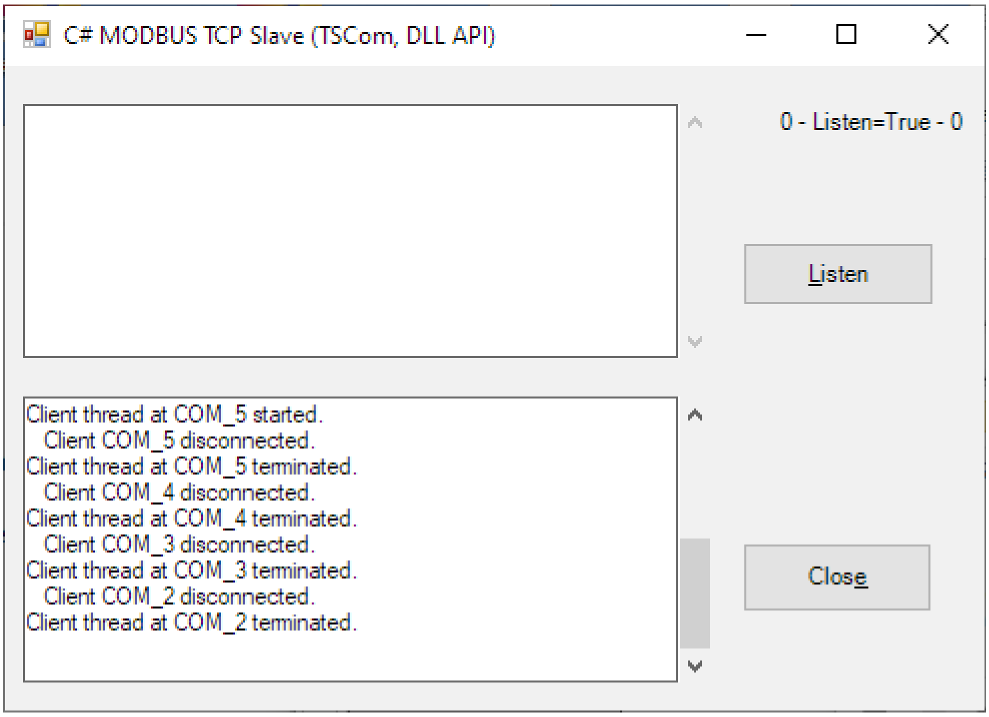
Modbus slave with C / C++



C:\ADONTEC\Modbus\tstconmsl.exe

```
SuperCom MODBUS TCP Server (slave) sample

Ready.
Total clients served  actually connected  on duty (sec)
--------------------  ------------------  -------------
         7                     7                29.548
```

Modbus slave with C#

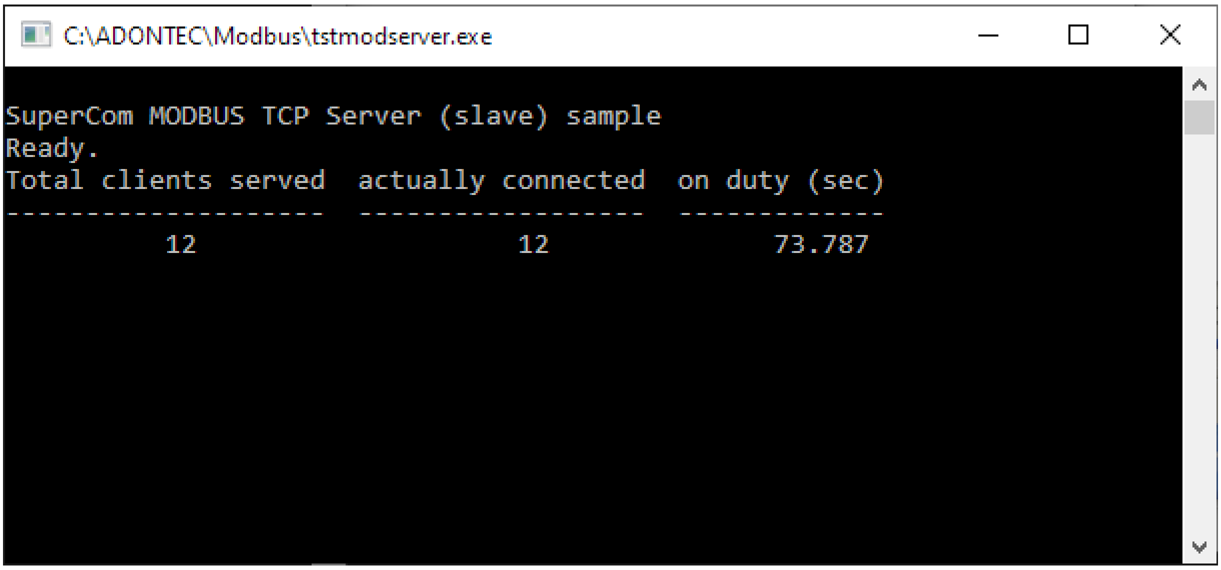


C# MODBUS TCP Slave (TSCom, DLL API)

0 - Listen=True - 0

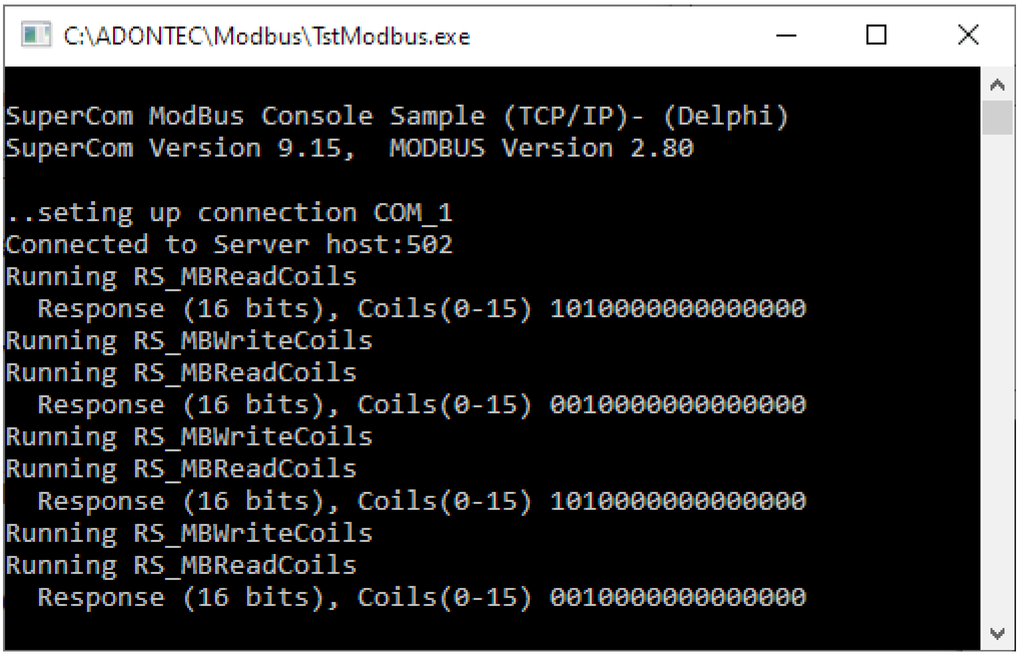
```
Client thread at COM_5 started.
  Client COM_5 disconnected.
Client thread at COM_5 terminated.
  Client COM_4 disconnected.
Client thread at COM_4 terminated.
  Client COM_3 disconnected.
Client thread at COM_3 terminated.
  Client COM_2 disconnected.
Client thread at COM_2 terminated.
```

Modbus slave with Delphi / Pascal



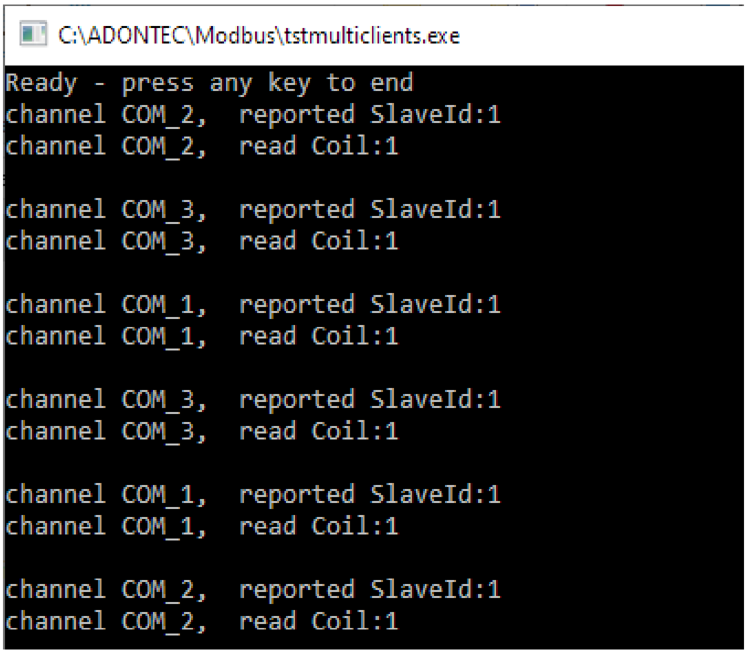
```
SuperCom MODBUS TCP Server (slave) sample
Ready.
Total clients served  actually connected  on duty (sec)
--------------------  ------------------  -------------
         12                   12                73.787
```

Modbus client with Delphi / Pascal



```
SuperCom ModBus Console Sample (TCP/IP)- (Delphi)
SuperCom Version 9.15,  MODBUS Version 2.80

..seting up connection COM_1
Connected to Server host:502
Running RS_MBReadCoils
  Response (16 bits), Coils(0-15) 1010000000000000
Running RS_MBWriteCoils
Running RS_MBReadCoils
  Response (16 bits), Coils(0-15) 0010000000000000
Running RS_MBWriteCoils
Running RS_MBReadCoils
  Response (16 bits), Coils(0-15) 1010000000000000
Running RS_MBWriteCoils
Running RS_MBReadCoils
  Response (16 bits), Coils(0-15) 0010000000000000
```

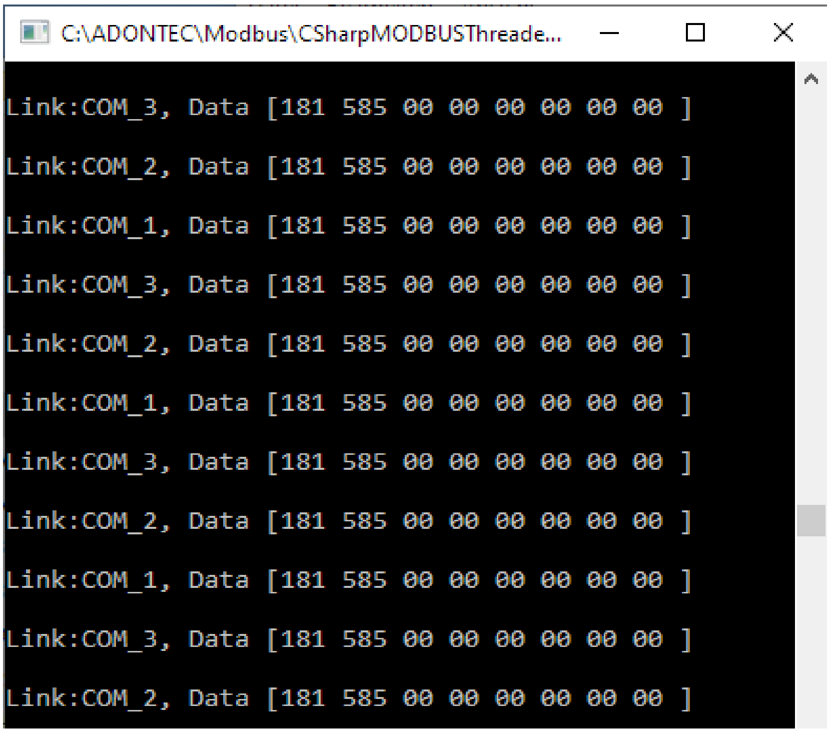
Modbus client with Delphi / Pascal
handling three connections simultaneously



C:\ADONTEC\Modbus\tstmulticlients.exe

```
Ready - press any key to end
channel COM_2,  reported SlaveId:1
channel COM_2,  read Coil:1

channel COM_3,  reported SlaveId:1
channel COM_3,  read Coil:1

channel COM_1,  reported SlaveId:1
channel COM_1,  read Coil:1

channel COM_3,  reported SlaveId:1
channel COM_3,  read Coil:1

channel COM_1,  reported SlaveId:1
channel COM_1,  read Coil:1

channel COM_2,  reported SlaveId:1
channel COM_2,  read Coil:1
```

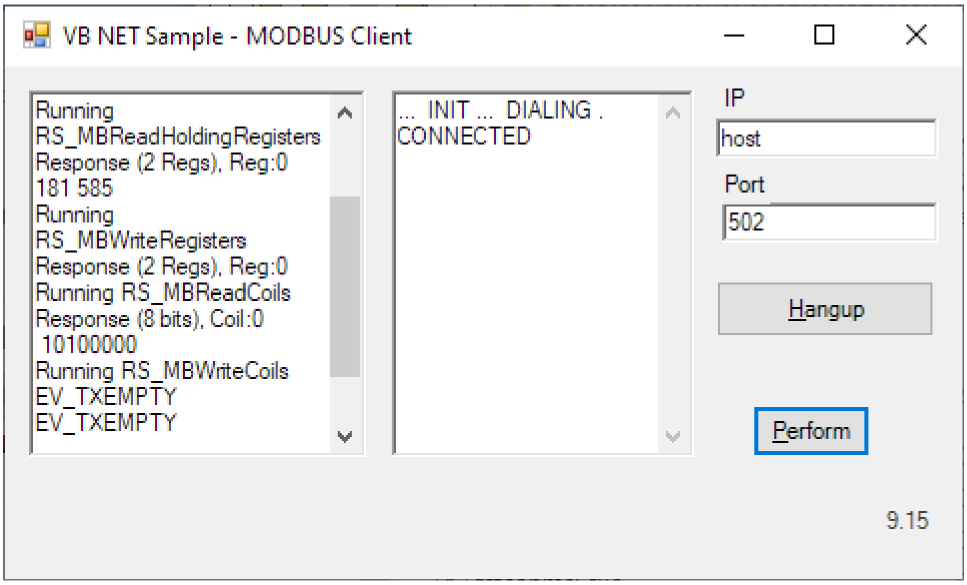
Modbus client with C#
handling three connections simultaneously and reading holding registers



C:\ADONTEC\Modbus\CSharpMODBUSThreade...

```
Link:COM_3, Data [181 585 00 00 00 00 00 00 ]

Link:COM_2, Data [181 585 00 00 00 00 00 00 ]

Link:COM_1, Data [181 585 00 00 00 00 00 00 ]

Link:COM_3, Data [181 585 00 00 00 00 00 00 ]

Link:COM_2, Data [181 585 00 00 00 00 00 00 ]

Link:COM_1, Data [181 585 00 00 00 00 00 00 ]

Link:COM_3, Data [181 585 00 00 00 00 00 00 ]

Link:COM_2, Data [181 585 00 00 00 00 00 00 ]

Link:COM_1, Data [181 585 00 00 00 00 00 00 ]

Link:COM_3, Data [181 585 00 00 00 00 00 00 ]

Link:COM_2, Data [181 585 00 00 00 00 00 00 ]
```

Modbus client with VB net



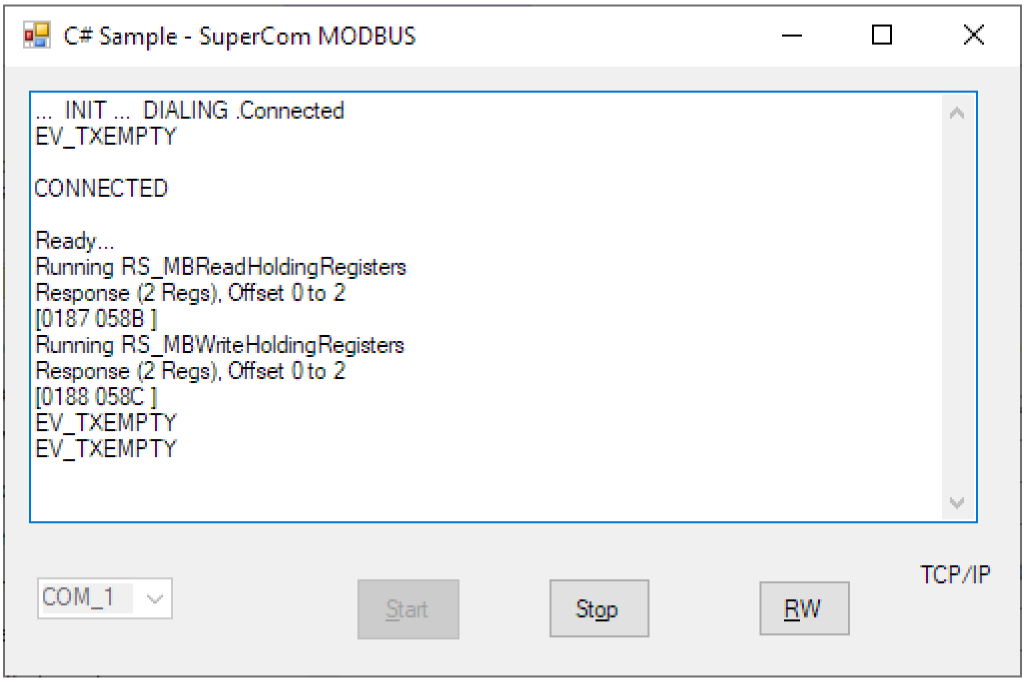
```
Running
RS_MBReadHoldingRegisters
Response (2 Regs), Reg:0
181 585
Running
RS_MBWriteRegisters
Response (2 Regs), Reg:0
Running RS_MBReadCoils
Response (8 bits), Coil:0
 10100000
Running RS_MBWriteCoils
EV_TXEMPTY
EV_TXEMPTY
```

```
... INIT ... DIALING .
CONNECTED
```

IP: host

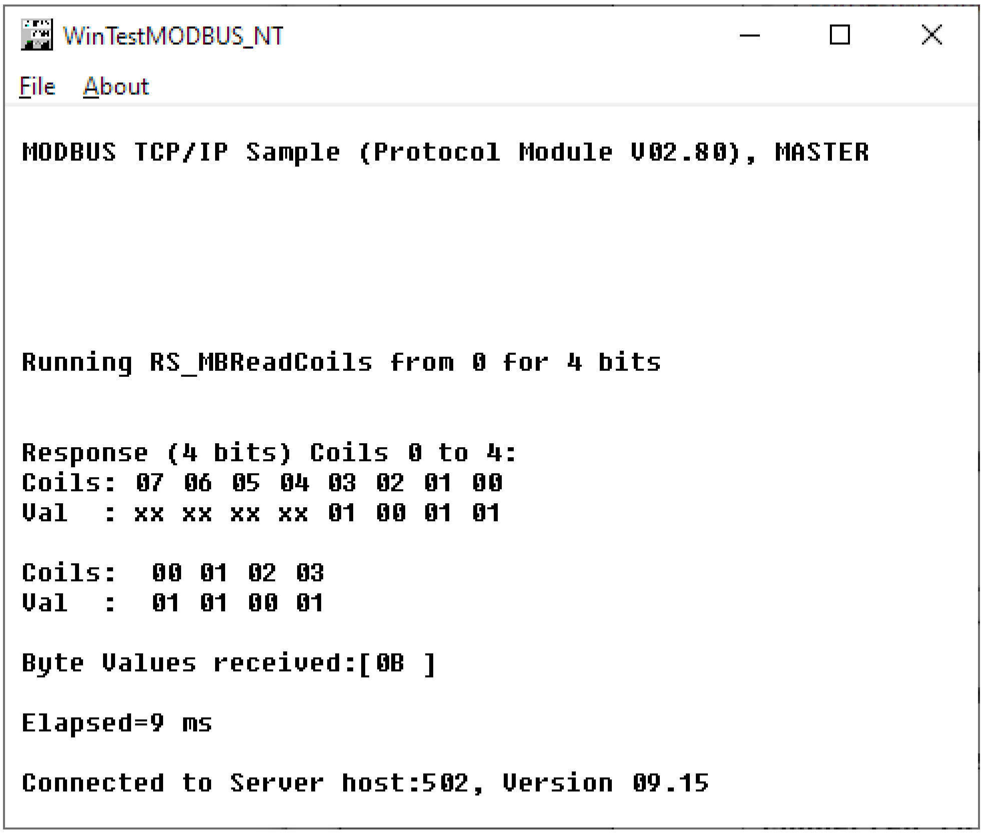
Port: 502

Modbus client with C#



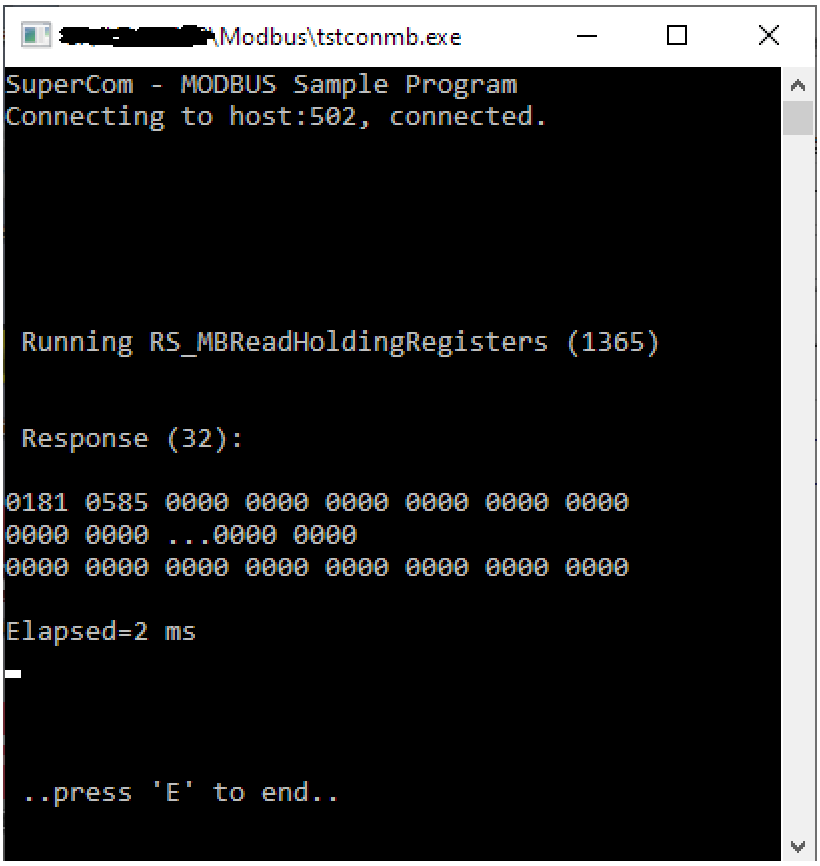
```
... INIT ... DIALING .Connected
EV_TXEMPTY

CONNECTED

Ready...
Running RS_MBReadHoldingRegisters
Response (2 Regs), Offset 0 to 2
[0187 058B ]
Running RS_MBWriteHoldingRegisters
Response (2 Regs), Offset 0 to 2
[0188 058C ]
EV_TXEMPTY
EV_TXEMPTY
```

Modbus client with C / C++



```
MODBUS TCP/IP Sample (Protocol Module V02.80), MASTER




Running RS_MBReadCoils from 0 for 4 bits


Response (4 bits) Coils 0 to 4:
Coils: 07 06 05 04 03 02 01 00
Val  : xx xx xx xx 01 00 01 01

Coils:  00 01 02 03
Val  :  01 01 00 01

Byte Values received:[0B ]

Elapsed=9 ms

Connected to Server host:502, Version 09.15
```

Modbus client with C / C++



```
SuperCom - MODBUS Sample Program
Connecting to host:502, connected.





 Running RS_MBReadHoldingRegisters (1365)


 Response (32):

0181 0585 0000 0000 0000 0000 0000 0000
0000 0000 ...0000 0000
0000 0000 0000 0000 0000 0000 0000 0000

Elapsed=2 ms




 ..press 'E' to end..
```

Summary

The **SuperCom MODBUS Protocol Library** provides a rock solid foundation to develop fast robust MODBUS capable applications.
The SuperCom MODBUS Protocol Library hides the complex MODBUS protocol offering one easy to use set of functions that can communicate data packets over serial and TCP/IP connections thus saving valuable time, reducing costs and ensuring quality results.

The SuperCom MODBUS Protocol Library supports data communication between devices connected to a serial port, on a bus system or network (TCP/IP). The protocol module supports ASCII and RTU (Remote Terminal Unit) operation mode (ASCII mode transfers ASCII codes and RTU binary data bytes in binary mode).

Run up to 255 simultaneous connections with Modbus devices.

There is only one API to learn! The same functions and parameters with any type of connection supported by SuperCom (e.g. Serial, TCP/IP, Modem, ISDN) and operation mode (Modbus ASCII, Modbus RTU, Modbus TCP).

Supported compilers
C#, C++, Delphi, Visual C++, Visual Basic, Visual Basic NET, C++ Builder, Borland C/C++, Microsoft C/C++, MinGW, Borland Pascal, Java, LabVIEW, PowerBuilder, PureBasic, VBA and other Windows programming tools (MS .NET ?).

Samples
for C/C++, C#, Visual C++, C++ Builder, Java, Pascal/Delphi, Visual Basic, Visual Basic NET (VB .NET), LabVIEW, PureBasic, ...

The SuperCom Suite contains more example programs and especially Modbus Slaves samples supporting multiple clients.

Speed
SuperCom is a very fast data communication library having very low latency and therefore suitable for time critical application like 9-Bit data comm or industrial real-time data comm.

But speed is relatively. The above tests were performed on relatively slow hardware on Desktop computer with Windows XP or Windows 7 or Notebook. On newer CPU it is expected to perform even better.

The speed (e.g. 2 ms per request) is also related to the performance of the remote device and in case of TCP/IP also the amount of switches, routers that lie in between.