# SuperCom - MODBUS Protocol Library

for Windows and Linux

Version 4.0

- Price List
- Product Range
- Call Me Back
- Translate

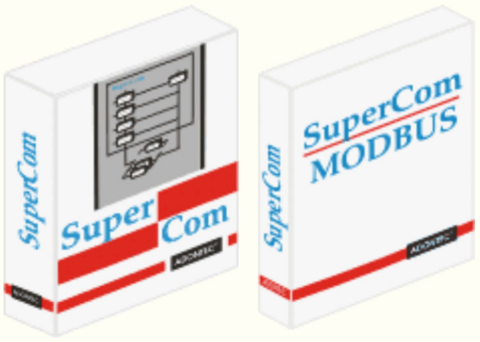
x64 1010010 32 Bit and 64 Bit Versions available!

## SuperCom MODBUS Protocol Library

MODBUS is an industrial data communication protocol used to control PLC's and automation machines connected through serial lines (RS-232, RS-422, RS-485) or Ethernet TCP/IP connections. The **SuperCom MODBUS Protocol Library** supports send and receive data with one or more devices which respect the MODBUS protocol.

The **SuperCom MODBUS Protocol Library** is the most complete MODBUS library and 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 (API) that communicate data packets over serial and TCP/IP connections thus saving valuable time, reducing costs and ensuring quality results.

Perform easily client or server device functionality in you application written for example with C, C++, C#, Delphi, Visual Basic (incl. NET), Java, LabView etc. Many samples included.



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.

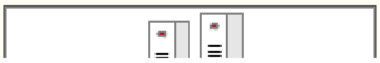
There is only one API to learn! The same functions used with Serial, TCP/IP or ISDN type of connections and operation mode (Modbus ASCII, Modbus RTU, Modbus TCP).

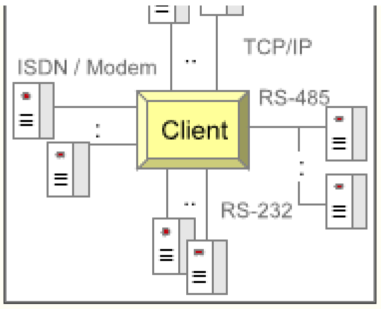
The SuperCom MODBUS Protocol Library also supports custom function codes and data packets by implementing functions that communicate transparently. Controlling machine specific extensions is realy easy using these functions.

**Accomplish with ease**

In most cases only a handful of functions are needed to talk to a Modbus capable PLC or controller. Your project is updated real fast. A lot of functions is backing you up to accomplish different tasks or configurations.

- Run a single or multiple **simultaneous connections** to modbus devices





(up to 255 connections per application).

- Runs smoothly even if multiple concurrently executing connections.
- Run multiple Modbus masters.
- Very short reaction times. Achieve transaction times of 2 ms per request. (See also PDF)
- Functions optimized for high data throughput.
- One portable common API across Windows, Linux, serial, TCP/IP etc.
- Connect one or more MODBUS RTU or ASCII clients to one or more Modbus servers.
- Modbus device simulation incl. samples.
- Functions optimized for high data throughput.
- A common and portable API for Windows, Linux, serial, TCP/IP, etc.
- Depending on the SuperCom software used, the SuperCom MODBUS protocol module can control multiple serial and/or TCP/IP connections simultaneously.
- A MODBUS server created with the SuperCom MODBUS Protocol Library can handle up to 254 client connections concurrently (samples included).
- The SuperCom MODBUS Protocol Library can control, based on the used SuperCom software, **serial and/or TCP/IP** connections.
- The SuperCom MODBUS Protocol Library can run concurrently connections thus polling more than one plc the same time is possible.
- Receive Modbus data packets and *simulate one or more MODBUS slaves or Protocol Gateway*.
- Create your own **Modbus Gateway** or **Bridge** to route data between serial interface and TCP/IP network, for example a *MODBUS TCP to MODBUS RTU* Gateway.
- Use SuperCom for Modbus bridging (e.g. serial to TCP/IP).
- Create your own **Protocol Gateway** using one of the other SuperCom industrial protocols (PLC protocols) offered or use your own custom protocol.
- Create your own **Modbus Slave** that can handle requests on its own or handed over to the application (manual mode) or mixed. A **Modbus Slave**, created with SuperCom, can support up to 254 concurrent client connections.
- Use with PLC from SIEMENS, Allen Bradley, Schneider, WAGO, other Modbus capable PLC, Modbus devices and controller. A pre-installed MODBUS protocol module on PLC or connected controller or device assumed.
- Receiving and transmitting of Modbus raw data is supported
- Low level **Data monitoring and recording** also supported.
- Supports NET 2.0 and higher and NET Core. More ...

The **SuperCom MODBUS Protocol Library** uses the *SuperCom Communication Layer* which provides a rock solid foundation to develop data communication software fast and without headache. Thus it makes no difference to the programmer if the MODBUS protocol used over TCP/IP or serial lines (RS-232, RS-422, RS-485, Modem, TAPI).

The **SuperCom MODBUS Protocol Library** accesses the Modbus remote station directly without using another software layer (e.g. OPC server or driver from third party) that can cause delays.

SuperCom enables the application to connect to a PLC (nearly from anywhere, e.g. Modem, TAPI, ISDN, Ethernet (TCP/IP) Internet, serial lines e.g. RS-232, RS-422, RS-485) and transfer data and/or read, write operands.

**Currently supported compiler**

As usual with SuperCom, many well-known compilers are supported. Others are constantly coming. The SuperCom MODBUS library can currently be used with the following compiler languages and compilers: C / C++, C#, Visual C++, C++ Builder, Java, Pascal, Delphi, LabVIEW, Visual Basic, Visual Basic .NET (VB .NET), VBA, PowerBuilder, PureBasic. If you miss one, please inquire. (See also)

## Samples - MODBUS Protocol API:

### 1. Read a single coil

C/C++

Init Sequence: ComInit RS_OpenLink class CSuperCom / CTcpClient

```
#define SLAVE_ID  1
TCOMMID Com = COM_2; // Com index, e.g. using serial port COM2

    // Serial, TCP, ... - native API
ComInit (Com);
ComSetState (Com, 9600, ...);

  :

-- Access --
RS_MBSetConfig(Com,,SuperCom.MODBUS.MODBUS_MODE_RTU,,);

if (RS_MBReadCoil (Com,
                   SLAVE_ID,
                   wCoil,
                   &Buffer))
{
    printf("Coil[%d] = %s ", wCoil, Buffer?"TRUE":"FALSE");
}
else
{
    int ErrorCode = RS_MBGetLastError(Com);

    if (ErrorCode == MB_ERR_EXCEPTION)
        printf("Exception %02X reported from server ", RS_MBGetException(Com));
    else
        printf("Error %d", ErrorCode);
}

ComReset(Com);
```

C/C++ C# Delphi Visual Basic

The above samples are nearly complete programs. More Init-Sequences can be found here.

### 2. Read/Write Registers

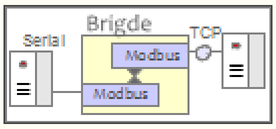
C/C++

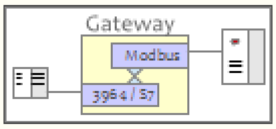
```
#define SLAVE_ID  1
```

```
    TCOMMID Com = COM_2; // Com index e.g. using serial port COM2

    WORD Buffer [10];
    WORD wStart=0x0000;
    WORD wCount=1;
    WORD wValue=0x0020;
         :
    -- Init Sequence (see above) --
         :
    RS_MBSetConfig(Com,,MODBUS_MODE_RTU,,);

    if (RS_MBWriteRegister(Com, SLAVE_ID, wStart, wValue))
        printf ("Read Success.\n");
    else
        printf ("Error: %d\n", RS_MBGetLastError(Com));

    if (RS_MBReadHoldingRegisters(Com,
                               SLAVE_ID,
                               wStart,
                               &wCount,
                               Buffer))
    {
        printf("Read %d Register:", wCount);
        for (int i=0; i<wCount; i++)  printf ("%4X",Buffer[i]);
    }
    else
        printf ("Error: %d\n", RS_MBGetLastError(Com));
```

C/C++ C# Delphi Visual Basic

## Capture, Store, Forward - MODBUS Slave or Gateway functions





Communication at lower data packet level is possible thus allowing to capture, store and forward of MODBUS data packets or transmit replies. Monitoring MODBUS data packets, simulating a *MODBUS Slave* or building a protocol converter / gateway is possible. *Simulating* a MODBUS Slave can be very handy while testing the software (slave examples available).

## Supported Protocols

The *SuperCom MODBUS Library* enables secure and stable data communications with different Modbus devices from different manufacturers. It supports almost all known Modbus options and variations.

The *SuperCom MODBUS Library* implements the standard MODBUS protocol for serial and TCP/IP connections* based on the official specification by the MODBUS organization. SuperCom supports **MODBUS ASCII** and **MODBUS RTU** protocol with serial interfaces (RS-232, RS-422, RS-485, Modem, TAPI) and the **MODBUS TCP/IP** (**MODBUS/TCP**) and **Open MODBUS TCP** with TCP/IP networks.
*A corresponding SuperCom license (Serial and/or TCP/IP) assumed.

The *SuperCom MODBUS Protocol Library* is a very complete MODBUS protocol stack!

Also included, non standard variants of the MODBUS protocol e.g. *RTU over IP* (also known as MODBUS RTU/IP, MODBUS RTU over TCP) enable the SuperCom application to talk to OMTS devices (OMTS = Out of the MODBUS TCP/IP specification). Since not an official specification, additional names and variants may exists.

Small *SuperCom MODBUS Server examples* can be found here.

## How to use?

The *SuperCom MODBUS Protocol Protocol Library* can be used over any *communication link* supported by SuperCom (currently Serial (RS-232, RS-422, RS-485, Modem, TAPI), TCP/IP, ISDN). For a list of available SuperCom packages containing MODBUS protocol, see the following chart.

The use of one common API for Serial, TCP/IP and ISDN applies here too (any SuperCom ActiveX API and/or DLL API).

## MODBUS Slave Simulator

Speeds up development process. The Modbus slave support included in this software also provides Modbus simulator software. Especially in the SuperCom Suite package. An easy to use and fast working test environment to deliver quality software. It also supports the creation of software based (virtual) Modbus devices for Modbus clients.

## What to order?

You will find promotional combinations with MODBUS (e.g. SuperCom Serial incl. MODBUS, SuperCom for TCP/IP incl. MODBUS, SuperCom Protocol Engine, SuperCom Suite).

The order code 638 or 638400 is needed only when the MODBUS protocol module is ordered later in addition to a SuperCom package. The most preferable way (incl. the best price) is buying one package from the above pricelist already including the MODBUS functionality.

## License Information

Executables developed using *SuperCom MODBUS Protocol Module* can be distributed royalty free. More ...

## 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.

**PDF Documents:**

- PDF document with more information and images of some of the included sample programs and some speed tests
- SuperCom-MODBUS-Software-Library

---

## What developers say

*... and the client is now polling real fast consuming low CPU. Great support.*
*Rob, USA.*

*I'm really impressed. Very sophisticated software that works fast and is flexible to use. Excelent work guys!*
*Pete, UK.*

*Great library, the GUI is not blocked.*
*Ch.., USA.*

*The slave application works like a charm.*
*Jon, Australia.*

*We were really surprised by the short reaction times and how smoothly it works even on multiple concurrent TCP connections.*
*M.., Poland.*

*Works faster than expected!*
*S.., France.*

*... issue solved, thanks.*
*P.., The Netherlands.*

---

MODBUS library | MODBUS RTU | MODBUS ASCII | MODBUS TCP | MODBUS over IP | MODBUS Serial Library | MODBUS RS-485 | MODBUS Slave | MODBUS sample C, C++, C#, Pascal, Delphi, Java, LabView, Visual Basic, VBA | MODBUS C library | MODBUS C++ library | MODBUS C# library | MODBUS Delphi library | MODBUS Pascal library | MODBUS VB library | MODBUS RTU example | MODBUS C Code Example | MODBUS slave simulator

---

Home Back

ADONTEC®

It Simply Works!

Updated on: 2022-09-07 08:22:22

Page generation time: 0.13 sec