



# Modbus RTU: Dynamic Configuration



When a Modbus COM port device is inserted in the tree, only static assignment of the parameters is possible.

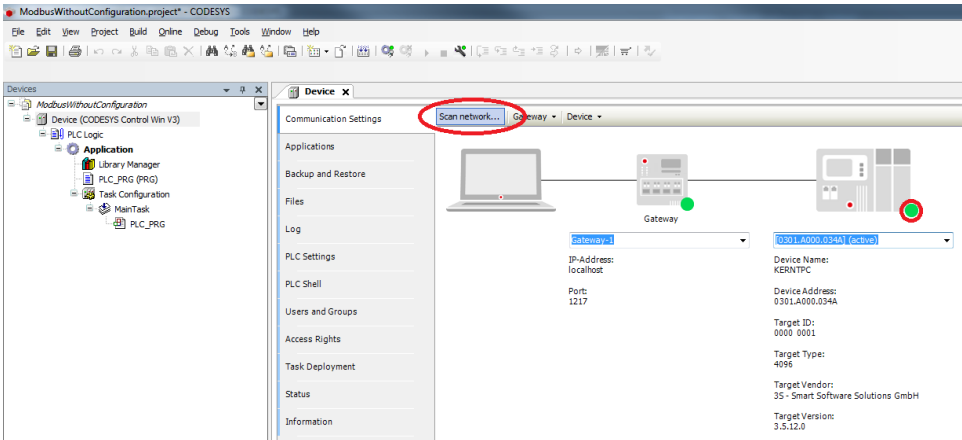
If it is necessary for the assignment to be dynamic, then the **complete** handling must be implemented in the IEC code



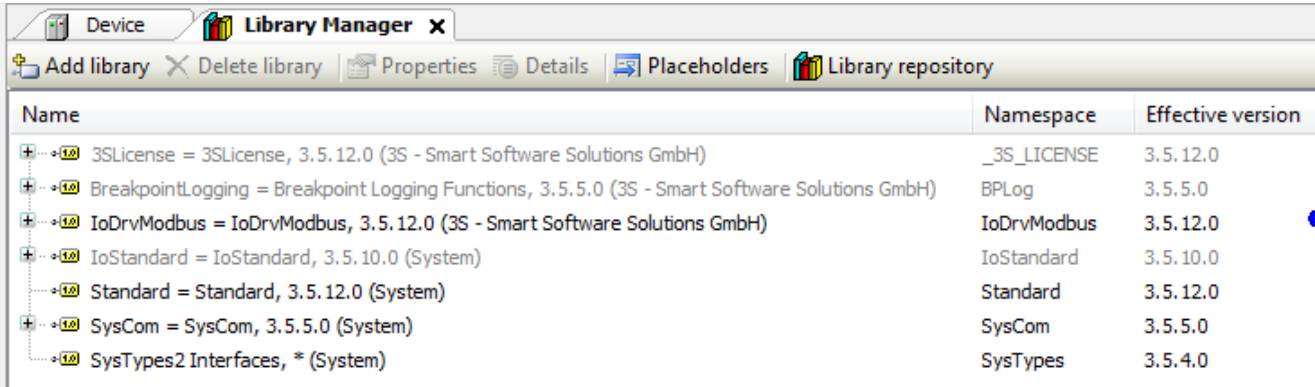
Please note that the I/O Manager does not synchronize the different processes in the dynamic configuration.

This can cause the update behavior of the variables to differ from the static configuration, especially if it is accessed from several tasks.

- Create a "Standard project" and select *CODESYS Control Win V3* as the device.
- Define the target system by means of the *Network scan*.



- Open the *Library Manager* and add the *Netzwerk* library.
  - *IoDrvModbus*
  - *SysCom*
  - *SysTypes2 Interfaces*



- Adapt the POU *PLC\_PRG* as follows:

Declaration

```

VAR
  xComPortOpen          : BOOL;
  xComPortError         : BOOL;
  rtsResult             : RTS_IEC_RESULT;
  stComPortSettings     : SysCom.ComSettings;
  hComPort              : RTS_IEC_HANDLE;

  awReadBuffer          : ARRAY[0..31] OF WORD;          //note: adjsut to your requirements, max.
is 128 Modbus Registers
  awWriteBuffer         : ARRAY[0..31] OF WORD;

  xExec                 : BOOL;
  fbModbusRequest       : IoDrvModbus.ModbusRequest2;
  eComError             : IoDrvModbus.MB_ErrorCodes;
END_VAR

```

#### Implementierung

```

IF(NOT xComPortOpen AND NOT xComPortError) THEN
  stComPortSettings.sPort := 3;
  stComPortSettings.byStopBits := 1;
  stComPortSettings.byParity := 0;          //EVEN:=2, ODD:=1 or NONE:=0*
  stComPortSettings.ulBaudRate := 19200;    //1200, 2400, 4800, ..., 115000 bps
  stComPortSettings.ulTimeout := 0;
  stComPortSettings.ulBufferSize := 256;    //Equals max size of Modbus packet

  hComPort := SysCom.SysComOpen2( pSettings := ADR(stComPortSettings), pSettingsEx := 0,
  pResult := ADR(rtsResult));

  xComPortError := (hComPort = RTS_INVALID_HANDLE OR rtsResult <> 0);
  xComPortOpen := NOT xComPortError;
END_IF

IF(xComPortOpen) THEN
  //apply new command
  fbModbusRequest.modbusCommand.uiFunctionCode := 23;          //Reda/Write Multiple Register
  fbModbusRequest.modbusCommand.uiReadOffset := 0;
  fbModbusRequest.modbusCommand.uiReadLen := 2;
  fbModbusRequest.modbusCommand.uiWriteOffset := 0;
  fbModbusRequest.modbusCommand.uiWriteLen := 2;

  fbModbusRequest.pRecvData := ADR(awReadBuffer);
  fbModbusRequest.pSendData := ADR(awWriteBuffer);

  fbModbusRequest.tResponseTimeout := T#500MS;

  //note: if different tasks access the IO-Buffers then use some intermediate buffers
  //      that are thread safe (use e.g mutex or semaphore) before calling the FB
  fbModbusRequest(
    hComPort := hComPort,
    xExecute := xExec,
    usiSlaveAddr := 2,
    byModbusError => eComError);

  IF(fbModbusRequest.xDone) THEN
    //here you get valid data
    IF(awWriteBuffer[0] = awReadBuffer[0]) THEN
      ;
    END_IF
  ELSIF(fbModbusRequest.xError) THEN
    IF(eComError = MB_ErrorCodes.RESPONSE_TIMEOUT) THEN          //no cable plugged, wrong Com-
Port settings ?
      ;
    END_IF
  END_IF
END_IF
END_IF

```

- Start the project and test the functionality.