Instruction SNQ: Modbus TCP

1 Purpose of this Document

This document describes the SNQ's Modbus TCP/UDP interface. It is intended to provide the necessary information to integrate our systems as Modbus TCP/UDP servers into a network. However, basic knowledge about networking (IP, TCP, UDP, Modbus) is required.

2 Definitions

SNQ	NOxController (control an dosing unit)
DPCU	Double Pump Control Unit (supplies a group of SNQs with reactant)

3 Network Overview

Our complete system consists of a maximum of 16 SNQ and 1 DPCU. We use the network address range 192.168.80.0/24. If we have a connection to an external network, we use a router. A fixed IP address of the external network is assigned to the interface "LAN1" of the router. "LAN2" belongs to our network, the assigned address is 192.168.80.1.

Our router performs Destination Network Address Translation (DNAT) by default. The table below shows the assignment.

Unit	IP address	local port	DNAT port
DPCU	192.168.80.32	502	50003
SNQ 1	192.168.80.40	502	50103
SNQ 2	192.168.80.48	502	50203
SNQ 3	192.168.80.56	502	50303
SNQ 4	192.168.80.64	502	50403
SNQ 5	192.168.80.72	502	50503
SNQ 6	192.168.80.80	502	50603
SNQ 7	192.168.80.88	502	50703
SNQ 8	192.168.80.96	502	50803
SNQ 9	192.168.80.104	502	50903
SNQ 10	192.168.80.112	502	51003
SNQ 11	192.168.80.120	502	51103
SNQ 12	192.168.80.128	502	51203
SNQ 13	192.168.80.136	502	51303
SNQ 14	192.168.80.144	502	51403
SNQ 15	192.168.80.152	502	51503
SNQ 16	192.168.80.160	502	51603

Example: The router's "LAN1" has the address 10.0.12.128. If you want to address the Modbus TCP server of SNQ 1, use the IP address 10.0.12.128 and the TCP port 50103 (10.0.12.128:50103). Our router will forward this to SNQ 1 (192.168.80.40:502).

4 Modbus

4.1 General

The Modbus Register Map is found in the document C.01763 (Hug Engineering AG).

Our systems provide a Modbus server. They support the Modbus function code 3 (read multiple holding registers). The Modbus protocol can be used either over TCP or UDP.

Each of our system provides an array of 16 holding registers. By definition, holding registers have addresses between 40001 and 49999. A potential source of confusion is the relationship between the reference numbers used in Modbus functions, and the register numbers used in Modicon PLCs. For historical reasons, user reference numbers were expressed as decimal numbers with a starting offset of 1. However, Modbus uses the more natural software interpretation of an unsigned integer index starting at zero. So a Modbus message requesting the read of a register at offset 0 would return the value known to the application programmer as found in register 40001 (memory type 4 = holding register, reference 0001).

MODBUS uses a big-endian representation for addresses and data items. This means that in a register (consisting of 2 bytes) the most significant byte is at the left side, the least significant byte at the right side (assuming that the memory array is from left to right).

The unit identifier (byte 6 of the Modbus request) has to be 1. Requests with unit identifiers other than 1 are not responded.

Modbus TCP and Modbus UDP are supported.

4.2 Format of the Modbus Request (fc3)

Byte 0: transaction identifier – copied by server – usually 0
Byte 1: transaction identifier – copied by server – usually 0
Byte 2: protocol identifier = 0
Byte 3: protocol identifier = 0
Byte 4: length in bytes (high byte) = 0 (All messages are smaller than 256 kB.)
Byte 5: length in bytes (low byte) = number of bytes following
Byte 6: unit identifier = 1 (We use 1 for our systems)
Byte 7: function code = 3 (fc3 = read holding registers)
Byte 8: start reference number (high byte)
Byte 9: start reference number (low byte) (0 corresponds to Modbus register 40001.)
Byte 10: number of words (registers) to be read (high byte)
Byte 11: number of words (registers) to be read (low byte)

4.3 Format of the Modbus Response (fc3)

Byte 0: transaction identifier – copied by server – usually 0
Byte 1: transaction identifier – copied by server – usually 0
Byte 2: protocol identifier = 0
Byte 3: protocol identifier = 0
Byte 4: length in bytes (high byte) = 0 (All messages are smaller than 256 kB.)
Byte 5: length in bytes (low byte) = number of bytes following

Byte 6: unit identifier = 1 (We use 1 for our systems)
Byte 7: function code = 3 (fc3 = read holding registers)
Byte 8: length (number of bytes following)
Byte 9: first read register (high byte)
Byte 10: first read register (low byte)
Byte 11: second read register (high byte)
Byte 12: second read register (low byte)
etc.

4.4 Example

The following example shows two complete Ethernet frames – the first with a Modbus TCP request (query), the second with the corresponding response. Ethernet, IP, TCP and Modbus are highlighted.

```
No.     Time                        Source                Destination           Protocol Info
      4 2009-08-13 15:47:04.162988 192.168.21.8          192.168.80.40         Modbus/TCP query
[ 1 pkt(s)]: trans:     0; unit:   1, func:   3: Read multiple registers.

0000   00 90 e8 0d 85 15 00 02 b3 ac 0a 81 08 00 45 00   ..............E.
0010   00 34 38 e5 40 00 40 06 1b 5e c0 a8 15 08 c0 a8   .48.@.@..^......
0020   50 28 84 b5 01 f6 70 af d9 72 73 f2 e2 02 50 18   P(....p..rs...P.
0030   16 d0 8a 93 00 00 00 00 00 00 06 01 03 00 00      ...............
0040   00 10                                             ..

No.     Time                        Source                Destination           Protocol Info
      5 2009-08-13 15:47:04.182551 192.168.80.40         192.168.21.8          Modbus/TCP response
[ 1 pkt(s)]: trans:     0; unit:   1, func:   3: Read multiple registers.

0000   00 02 b3 ac 0a 81 00 90 e8 0d 85 15 08 00 45 00   ..............E.
0010   00 51 9f 49 00 00 3f 06 f5 dc c0 a8 50 28 c0 a8   .Q.I..?.....P(..
0020   15 08 01 f6 84 b5 73 f2 e2 02 70 af d9 7e 50 18   ......s...p..~P.
0030   16 d0 ee 75 00 00 00 00 00 00 23 01 03 20 00      ...u......#.. .
0040   0f 00 00 80 00 80 00 80 00 ff 6a ff 6a ff e7 00   ..........j.j...
0050   00 00 00 80 00 80 00 00 00 00 00 12 34 56 78      ............4Vx
```

5 Links to Related Documents

http://wingpath.co.uk/docs/modbus_tcp_specification.pdf. This document defines the Modbus TCP protocol, which is used to send Modbus messages over a network using the TCP/IP protocols. It also contains some intelligent commentary on the Modbus protocol, and an appendix on client and server implementation, which may be useful if this area is new to you.

http://modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf. This is the current definition of Modbus function codes and the format of the corresponding request and response messages. It does not cover how these messages are packaged in the RTU, ASCII or TCP variants of the protocol.

http://modbus.org/docs/Modbus_Messaging_Implementation_Guide_V1_0b.pdf. This consists mainly of what appear to be design notes for Schneider Automation's implementation of Modbus TCP. You will probably find it hard to read if you don't already have a good understanding of the area, and not very useful if you do. It is, however, the only document on the modbus.org web-site that contains (in section 3.1) a description of the Modbus TCP protocol.