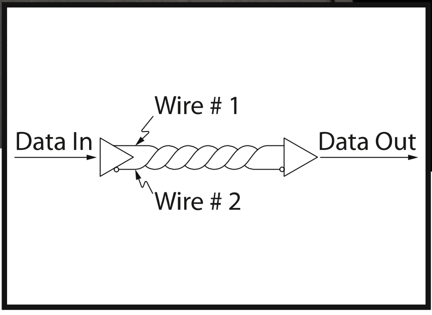
MAKING MODERN LIVING POSSIBLE



Modbus RTU

Communication Setup

Manual

Introduction

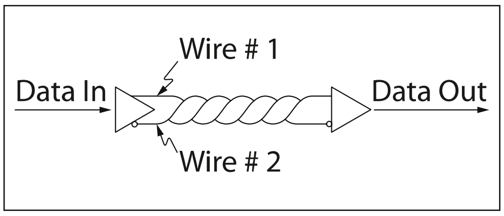
This manual explains how to physically establish and configure the communication between a Performer VSD and a system controller, using the Modbus RTU (Remote Terminal Unit) protocol.
For detailed information about Modbus communication, refer to Modbus Application Protocol Specifications V1.1b from www.modbus.org

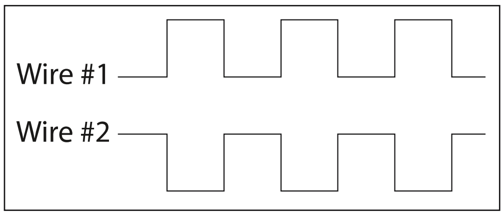
Modbus specification

The system controller is connected to the variable speed drive via a RS-485 communication network.

RS-485 Communication

The Electronics Industry Association (EIA) established the RS-485 standard as a guide for developing a multi-drop, bi-directional communication network.
RS-485 systems can be implemented using two-wires or four-wires modes. Danfoss uses the two-wires system with ground shield. With the two-wires system, communication is half-duplex (cannot transmit and receive at the same time).
Each signal uses one twisted–pair line – two wires twisted around themselves. This is known as balanced data transmission. The signal on one wire is ideally the exact opposite of the signal on the second wire. In other words, if one wire is transmitting a high, the other wire will be transmitting the low, and vice versa.
Since RS-485 is a multipoint communication system, all devices are connected to the single twisted-pair cable.





The RS-485 system uses master/slave architecture, where each slave device (Performer VSD) has its unique address and responds only to packets addressed to this device. The packets are generated by the master (system controller), which periodically polls all connected slave devices. Data travels over the single line in both directions.
All of the devices must have drivers with tri-state outputs (including the master). Tri-state (or three-state) buffers allow multiple devices to drive the same output; in this case, the output is the twisted-pair cable they all share. They are called tri-state because they have three output states: high (1), low (0), and high impedance (Z). The high-impedance state effectively isolates the device output from the cable when it is not transmitting.
A basic RS-485 system requires an I/O driver with differential outputs and an I/O receiver with differential inputs. Since the signal is transferred via a twisted pair of wires, if noise or interference is introduced into the line, the voltage difference (between twisted pair wires) of this interference is almost zero. Because the input to the receiver is differential, this interference is eliminated. Differential inputs also ignore different earth potentials of the transmitter and the receiver.
For correct operation of the transmitter and the receiver, a return signal between the grounding of individual devices is required. The ground shield is used for this purpose.

RS-485 specifications

Specifications		RS-485
Mode of operation		Differential
Total number of drivers and receivers		1 driver 32 receivers
Maximum driver output signal		-7V to +12V
Driver output signal level (Loaded min)	Loaded	+/- 1.5V
Driver output signal level (Unloaded max)	Unloaded	+/- 6V
Driver load impedance (Ohm)		54
Max driver current in high Z state	Power on Power off	+/- 100µA +/- 100µA
Slew rate (max)		n/a
Receiver input voltage range		-7V to +12V
Receiver input sensitivity		+/- 200mV
Receiver input resistance (Ohm)		>= 12k

Cable specification

When choosing a transmission line for RS-485, it is necessary to examine the required distance of the cable and the data rate of the system. Losses in a transmission line are a combination of AC losses (skin effect), DC conductor loss, leakage, and AC losses in the dielectric. In high-quality cable, the conductor losses and the dielectric losses are on the same order of magnitude.
The recommended maximum Modbus cable length between the Compressor and the system controller should not exceed 1200 meters (4000 feet).
While the RS-485 specification does not specify cabling, the recommendation is 24AWG shielded twisted-pair cable with a shunt capacitance of 16 pF/ft and 100Ω impedance.
Another choice is the same cable commonly used in the twisted-pair Ethernet cabling. This cable is commonly referred to as Category 5 cable. The cable has a maximum capacitance of 17 pF/ft (14.5 pF/ft typical) and characteristic impedance of 100Ω.

Cable connection

Connect the RS-485 cable to the dedicated terminals of the Variable Speed Drive:
Positive (+) polarity wire to terminal 68
Negative (-) polarity wire to terminal 69
Cable screen to cable clamp
The other end of the RS-485 cable to the system controller (master) ensuring the polarity of the wires matches with the polarity of the Variable Speed Drive terminals

Modbus RTU message

CDS302 with Modbus RTU

The controllers are set up to communicate on the Modbus network using RTU (Remote Terminal Unit) mode, with each byte in a message containing two 4-bit hexadecimal characters. The format for each byte is shown below.

Start bit	Data Byte								Stop / parity	Stop

Coding System	8-bit binary, hexadecimal 0-9, A-F. Two hexadecimal characters contained in each 8-bit field of the message
Bits Per Byte	1 start bit
	8 data bits, least significant bit sent first
	1 bit for even/odd parity; no bit for no parity
	1 stop bit if parity is used; 2 bits if no parity
Error Check Field	Cyclical Redundancy Check (CRC)

Modbus RTU message structure

The transmitting device places a Modbus RTU message into a frame with a known beginning and ending point. This allows receiving devices to begin at the start of the message, read the address portion, determine which device is addressed (or all devices, if the message is broadcast), and to recognise when the message is completed. Partial messages are detected and errors set as a result. Characters for transmission must be in hexadecimal 00 to FF format in each field. The frequency converter continuously monitors the network bus, also during 'silent' intervals. When the first field (the address field) is received, each frequency converter or device decodes it to determine which device is being addressed. Modbus RTU messages addressed to zero are broadcast messages. No response is permitted for broadcast messages. A typical message frame is shown below.

Typical Modbus RTU message structure

Start	Address	Function	Data	CRC check	End
T1-T2-T3-T4	8 bits	8 bits	N x 8 bits	16 bits	T1-T2-T3-T4

Start / Stop field

Messages start with a silent period of at least 3.5 character intervals. This is implemented as a multiple of character intervals at the selected network baud rate (shown as Start T1-T2-T3-T4). The first field to be transmitted is the device address. Following the last transmitted character, a similar period of at least 3.5 character intervals marks the end of the message. A new message can begin after this period. The entire message frame must be transmitted as a continuous stream. If a silent period of more than 1.5 character intervals occurs before completion of the frame, the receiving device flushes the incomplete message and assumes that the next byte will be the address field of a new message. Similarly, if a new message begins prior to 3.5 character intervals after a previous message, the receiving device will consider it a continuation of the previous message. This will cause a time-out (no response from the slave), since the value in the final CRC field will not be valid for the combined messages.

Address field

The address field of a message frame contains 8 bits. Valid slave device addresses are in the range of 0 – 247 decimal. The individual slave devices are assigned addresses in the range of 1 – 247. (0 is reserved for broadcast mode, which all slaves recognize.) A master addresses a slave by placing the slave address in the address field of the message. When the slave sends its response, it places its own address in this address field to let the master know which slave is responding.

Function field

The function field of a message frame contains 8 bits. Valid codes are in the range of 1-FF. Function fields are used to send messages between master and slave. When a message is sent from a master to a slave device, the function code field tells the slave what kind of action to perform. When the slave responds to the master, it uses the function code field to indicate either a normal (error-free) response, or that some kind of error occurred (called an exception response). For a normal response, the slave simply echoes the original function code. For an exception response, the slave returns a code that is equivalent to the original function code with its most significant bit set to logic 1. In addition, the slave places a unique code into the data field of the response message. This tells the master what kind of error occurred, or the reason for the exception. Please also refer to the sections Function Codes Supported by Modbus RTU and Exception Codes.

Data field

The data field is constructed using sets of two hexadecimal digits, in the range of 00 to FF hexadecimal. These are made up of one RTU character. The data field of messages sent from a master to slave device contains additional information which the slave must use to take the action defined by the function code. This can include items such as coil or register addresses, the quantity of items to be handled, and the count of actual data bytes in the field.

CRC check field

Messages include an error-checking field, operating on the basis of a Cyclical Redundancy Check (CRC) method. The CRC field checks the contents of the entire message. It is applied regardless of any parity check method used for the individual characters of the message. The CRC value is calculated by the transmitting device, which appends the CRC as the last field in the message. The receiving device re-calculates a CRC during receipt of the message and compares the calculated value to the actual value received in the CRC field. If the two values are unequal, a bus time-out results. The error-checking field contains a 16-bit binary value implemented as two 8-bit bytes. When this is done, the low-order byte of the field is appended first, followed by the high-order byte. The CRC high-order byte is the last byte sent in the message.

Coils register addressing

In Modbus, all data are organized in coils and holding registers. Coils hold a single bit, whereas holding registers hold a 2-byte word (i.e. 16 bits). All data addresses in Modbus messages are referenced to zero. The first occurrence of a data item is addressed as item number zero. For example: The coil known as 'coil 1' in a programmable controller is addressed as coil 0000 in the data address field of a Modbus message. Coil 127 decimal is addressed as coil 007EHEX (126 decimal).

Coil Number	Description	Signal Direction
1-16	Frequency converter control word (see table below)	Master to slave
17-32	Frequency converter speed or set-point reference Range 0x0 – 0xFFFF (-200% ...~200%)	Master to slave
33-48	Frequency converter status word (see table below)	Slave to master
49-64	Open loop mode: Frequency converter output frequency Closed loop mode: Frequency converter feedback signal	Slave to master
65	Parameter write control (master to slave) 0 = Parameter changes are written to the RAM of the frequency converter 1 = Parameter changes are written to the RAM and EEPROM of the frequency converter.	Master to slave
66-65536	Reserved	

Frequency converter status word

Coil	0	1
01	Preset reference LSB	
02	Preset reference MSB	
03	DC brake	No DC brake
04	Coast stop	No coast stop
05	Quick stop	No quick stop
06	Freeze freq.	No freeze freq.
07	Ramp stop	Start
08	No reset	Reset
09	No jog	Jog
10	Ramp 1	Ramp 2
11	Data not valid	Data valid
12	Relay 1 off	Relay 1 on
13	Relay 2 off	Relay 2 on
14	Set up LSB	
15	Set up MSB	
16	No reversing	Reversing
33	Control not ready	Control ready
34	Frequency converter not ready	Frequency converter ready
35	Coasting stop	Safety closed
36	No alarm	Alarm
37	Not used	Not used
38	Not used	Not used
39	Not used	Not used
40	No warning	Warning
41	Not at reference	At reference
42	Hand mode	Auto mode
43	Out of freq. range	In frequency range
44	Stopped	Running
45	Not used	Not used
46	No voltage warning	Voltage warning
47	Not in current limit	Current limit
48	No thermal warning	Thermal warning

Holding registers

Register number	Description
00001-00006	Reserved
00007	Last error code from an FC data object interface
00008	Reserved
00009	Parameter index
00010-00990	000 parameter group (parameters 001 through 099)
01000-01990	100 parameter group (parameters 100 through 199)
02000-02990	200 parameter group (parameters 200 through 299)
03000-03990	300 parameter group (parameters 300 through 399)
04000-04990	400 parameter group (parameters 400 through 499)
... ...	
49000-49990	4900 parameter group (parameters 4900 through 4999)
50000	Input data: Frequency converter control word register (CTW).
50010	Input data: Bus reference register (REF).
... ...	
50200	Output data: Frequency converter status word register (STW).
50210	Output data: Frequency converter main actual value register (MAV).

How to control the frequency converter

This section describes codes which can be used in the function and data fields of a Modbus RTU message. For a complete description of all the message fields please refer to the section Modbus RTU Message Framing Structure.

Function codes supported by Modbus RTU

Modbus RTU supports use of the following function codes in the function field of a message:

Function	Function Code
Read coils	1 hex
Read holding registers	3 hex
Write single coil	5 hex
Write single register	6 hex
Write multiple coils	F hex
Write multiple registers	10 hex
Get comm. event counter	B hex
Report slave ID	11 hex

Function	Function Code	Sub-function code	Sub-function
Diagnostics	8	1	Restart communication
		2	Return diagnostic register
		10	Clear counters and diagnostic register
		11	Return bus message count
		12	Return bus communication error count
		13	Return bus exception error count
		14	Return slave message count

Modbus exception code

For a full explanation of the structure of an exception code response, please refer to the section Modbus RTU Message Framing Structure, Function Field.

Code	Name	Meaning
1	Illegal function	The function code received in the query is not an allowable action for the server (or slave). This may be because the function code is only applicable to newer devices, and was not implemented in the unit selected. It could also indicate that the server (or slave) is in the wrong state to process a request of this type, for example because it is not configured and is being asked to return register values.
2	Illegal data address	The data address received in the query is not an allowable address for the server (or slave). More specifically, the combination of reference number and transfer length is invalid. For a controller with 100 registers, a request with offset 96 and length 4 would succeed, a request with offset 96 and length 5 will generate exception 02.
3	Illegal data value	A value contained in the query data field is not an allowable value for server (or slave). This indicates a fault in the structure of the remainder of a complex request, such as that the implied length is incorrect. It specifically does NOT mean that a data item submitted for storage in a register has a value outside the expectation of the application program, since the Modbus protocol is unaware of the significance of any particular value of any particular register.
4	Slave device failure	An unrecoverable error occurred while the server (or slave) was attempting to perform the requested action.

How to access parameters

Parameter handling

The PNU (Parameter Number) is translated from the register address contained in the Modbus read or write message.
The parameter number is translated to Modbus as (10 x parameter number) DECIMAL.

Storage of data

The Coil 65 decimal determines whether data written to the frequency converter are stored in EEPROM and RAM (coil 65 = 1) or only in RAM (coil 65 = 0).

IND

The array index is set in Holding Register 9 and used when accessing array parameters.

Text blocks

Parameters stored as text strings are accessed in the same way as the other parameters. The maximum text block size is 20 characters. If a read request for a parameter is for more characters than the parameter stores, the response is truncated. If the read request for a parameter is for fewer characters than the parameter stores, the response is space filled.

Conversion factor

The different attributes for each parameter can be seen in the section on factory settings. Since a parameter value can only be transferred as a whole number, a conversion factor must be used to transfer decimals. Please refer to the Parameters section.

Parameter values

Standard data types are int16, int32, uint8, uint16 and uint32. They are stored as 4x registers (40001 – 4FFFF). The parameters are read using function 03HEX "Read Holding Registers." Parameters are written using the function 6HEX "Preset Single Register" for 1 register (16 bits), and the function 10HEX "Preset Multiple Registers" for 2 registers (32 bits). Readable sizes range from 1 register (16 bits) up to 10 registers (20 characters).

Non standard data types are text strings and are stored as 4x registers (40001 – 4FFFF). The parameters are read using function 03HEX "Read Holding Registers" and written using function 10HEX "Preset Multiple Registers." Readable sizes range from 1 register (2 characters) up to 10 registers (20 characters).

Examples

The following examples illustrate various Modbus RTU commands. If an error occurs, please refer to the Exception Codes section.

Read coil status (01 HEX)

Description: this function reads the ON/OFF status of discrete outputs (coils) in the frequency converter. Broadcast is never supported for reads.

Query: the query message specifies the starting coil and quantity of coils to be read. Coil addresses start at zero, i.e. coil 33 is addressed as 32.
Example of a request to read coils 33-48 (Status Word) from slave device 01:

Field Name	Example (HEX)
Slave Address	01 (frequency converter address)
Function	01 (read coils)
Starting Address HI	00
Starting Address LO	20 (32 decimals)
No. of Points HI	00
No. of Points LO	10 (16 decimals)
Error Check (CRC)	-

Response: the coil status in the response message is packed as one coil per bit of the data field. Status is indicated as: 1 = ON; 0 = OFF. The LSB of the first data byte contains the coil addressed in the query. The other coils follow toward the high order end of this byte, and from 'low order to high order' in subsequent bytes.
If the returned coil quantity is not a multiple of eight, the remaining bits in the final data byte will be padded with zeros (toward the high order end of the byte). The Byte Count field specifies the number of complete bytes of data.

Field Name	Example (HEX)
Slave Address	01 (frequency converter address)
Function	01 (read coils)
Byte Count	02 (2 bytes of data)
Data (Coils 40-33)	07
Data (Coils 48-41)	06 (STW=0607hex)
Error Check (CRC)	-

Force/Write single coil (05 HEX)

Description: this function forces / writes a coil to either ON or OFF. When broadcast the function forces the same coil references in all attached slaves.

Query: the query message specifies the coil 65 (parameter write control) to be forced. Coil addresses start at zero, i.e. coil 65 is addressed as 64. Force Data
= 00 00HEX (OFF) or FF 00HEX (ON).

Field Name	Example (HEX)
Slave Address	01 (frequency converter address)
Function	05 (write single coil)
Coil Address HI	00
Coil Address LO	40 (coil no. 65)
Force Data HI	FF
Force Data LO	00 (FF 00 = ON)
Error Check (CRC)	-

Response: the normal response is an echo of the query, returned after the coil state has been forced.

Field Name	Example (HEX)
Slave Address	01
Function	05
Force Data HI	FF
Force Data LO	00
Quantity of Coils HI	00
Quantity of Coils LO	01
Error Check (CRC)	-

Force/Write multiple coils (0F HEX)

Description : this function forces each coil in a sequence of coils to either ON or OFF. When broadcast the function forces the same coil references in all attached slaves.

Query : The query message specifies the coils 17 to 32 (speed set-point) to be forced. Coil addresses start at zero, i.e. coil 17 is addressed as 16.

Field Name	Example (HEX)
Slave Address	01 (frequency converter address)
Function	0F (write multiple coils)
Coil Address HI	00
Coil Address LO	10 (coil address 17)
Quantity of Coils HI	00
Quantity of Coils LO	10 (16 coils)
Byte Count	02
Force Data HI (Coils 8-1)	20
Force Data LO (Coils 10-9)	00 (ref. = 2000hex)
Error Check (CRC)	-

Response : the normal response returns the slave address, function code, starting address, and quantity of coils forced.

Field Name	Example (HEX)
Slave Address	01 (frequency converter address)
Function	0F (write multiple coils)
Coil Address HI	00
Coil Address LO	10 (coil address 17)
Quantity of Coils HI	00
Quantity of Coils LO	10 (16 coils)
Error Check (CRC)	-

Read holding registers (03 HEX)

Description : this function reads the contents of holding registers in the slave.

Query : the query message specifies the starting register and quantity of registers to be read. Register addresses start at zero, i.e. registers 1-4 are addressed as 0-3.

Field Name	Example (HEX)
Slave Address	01
Function	03 (read holding registers)
Starting Address HI	00
Starting Address LO	00 (coil address 17)
No. of Points HI	00
No. of Points LO	03
Error Check (CRC)	-

Response : the register data in the response message are packed as two bytes per register, with the binary contents right justified within each byte. For each register, the first byte contains the high order bits and the second contains the low order bits.

Field Name	Example (HEX)
Slave Address	01
Function	03
Byte Count	06
Data HI (Register 40001)	55
Data LO (Register 40001)	AA
Data HI (Register 40002)	55
Data LO (Register 40002)	AA
Data HI (Register 40003)	55
Data LO (Register 40003)	AA
Error Check (CRC)	-

Preset single register (06HEX)

Description : this function presets a value into a single holding register.

Query: the query message specifies the register reference to be preset. Register addresses start at zero, i.e. register 1 is addressed as 0.

Field Name	Example (HEX)
Slave Address	01
Function	06
Register Address HI	00
Register Address LO	01
Preset Data HI	00
Preset Data LO	03
Error Check (CRC)	-

Response : the normal response is an echo of the query, returned after the register contents have been passed.

Field Name	Example (HEX)
Slave Address	01
Function	06
Register Address HI	00
Register Address LO	01
Preset Data HI	00
Preset Data LO	03
Error Check (CRC)	-

Preset multiple registers (10 HEX)

Description : this function presets values into a sequence of holding registers.

Query : the query message specifies the register references to be preset. Register addresses start at zero, i.e. register 1 is addressed as 0. Example of a request to preset two registers (set parameter 1-05 = 738 (7.38 A)):

Field Name	Example (HEX)
Slave Address	01
Function	10
Starting Address HI	04
Starting Address LO	19
No. of Registers HI	00
No. of registers LO	02
Byte Count	04
Write Data HI (Register 4: 1049)	00
Write Data LO (Register 4: 1049)	00
Write Data HI (Register 4: 1050)	02
Write Data LO (Register 4: 1050)	E2
Error Check (CRC)	-

Response : the normal response returns the slave address, function code, starting address, and quantity of registers preset.

Field Name	Example (HEX)
Slave Address	01
Function	10
Starting Address HI	04
Starting Address LO	19
No. of Registers HI	00
No. of registers LO	02
Error Check (CRC)	-

Parameter list

Examples of Modbus Holding Registers (= Parameter number x 10)

				CDS302-15/18kW	… …
Parameter designation	Parameter number	Customer access	Modbus Holding Register	VSH088-G	
Language	0-01	r/w	10	English	
Motor speed unit	0-02	r/w	20	RPM	
Regional settings	0-03	r/w	30	International	
… … …					
Configuration mode	1-00	r/w	1000	Speed open loop	
Control mode	1-01	read	1010	Flux sensorless	
… … …					

Refer to Performer VSD CDS302 Installation Manual (ref MG34M102) for the complete list and updated list of parameters for all compressor models.

Warnings/Alarms

Refer to Performer VSD CDS302 Installation Manual (ref MG34M102) for warnings and alarms messages.

The Danfoss product range for the refrigeration and air conditioning industry

Within refrigeration and air conditioning, Danfoss is a worldwide manufacturer with a leading position in industrial, commercial and supermarket refrigeration as well as air conditioning and climate solutions.

We focus on our core business of making quality products, components and systems that enhance performance and reduce total life cycle costs – the key to major savings.

Controls for Commercial Refrigeration

Controls for Industrial Refrigeration

Electronic Controls & Sensors

Industrial Automation

Household Compressors

Commercial Compressors

Sub-Assemblies

Thermostats

Brazed plate heat exchanger

We are offering a single source for one of the widest ranges of innovative refrigeration and air conditioning components and systems in the world. And, we back technical solutions with business solution to help your company reduce costs, streamline processes and achieve your business goals.

Danfoss A/S • www.danfoss.com

member of:

ASERCOM

www.asercom.org

Danfoss Commercial Compressors http://cc.danfoss.com