Operating instruction manual

SDMO

Controller

APM303-MODBUS

Firmware version 1.1.0.1

33530004501_0_1
18/07/2014

Contents

List of figures

1 - Introduction

This document describes how to use the RS485 link available on the **APM303** module. It comprises 3 parts:

- RS485 link and Modbus protocol,
- List of information available on the **APM303**,
- Example applications of the RS485 link with the **APM303**.

Note: this document is applicable to version 1.1.0.1 of the firmware.

2 - Reminder on some units

This paragraph recaps some concepts in information encoding.

bit : the smallest elementary unit of information representation, the bit adopts a value of 0 or 1
byte : is equal to 8 bits
word : is equal to 2 bytes or 16 bits
long : is equal to 2 words or 4 bytes or 32 bits
baud : unit of measurement of information transmission speed, also expressed in bits/second (*)
(*) In our case, measurements in bauds or in bit/s (bps) are equal, since the signal is bivalent (2 values; 0 or 1).

A byte is a set of 8 bits, written in the form; 1001 0001 (where each bit adopts the value 0 or 1). For each byte, that makes 256 different combinations. A word is a set of 16 bits, i.e. 65536 possible values.

In this document, when values or addresses are expressed in hexadecimal, they are always preceded by the symbol "0x". Values not preceded by any symbol are expressed in decimal.

3 - RS485 link and Modbus protocol

This means of communication available on the **APM303** comprises 2 essential parts:

Hardware The information is exchanged in the form of 0s and 1s. This hardware part defines how a 0 and a 1 are represented. It is the RS485 link which carries out this operation.

Software The organisation of the 0s and 1s received indicates what message a machine wants to convey to another machine. This "organisation" is known as the protocol. In our case, the Modbus protocol is used.

3.1 - RS485 link

The RS485 link is the physical medium for communication. It is a differential (or symmetric) link. It comprises 2 active wires (**A+** and **B-**) and shielding. This is a multipoint serial link. It interconnects several pieces of equipment. The maximum bus length is 1200 metres.

The RS485 bus requires a line impedance of 120 Ohms. The cable must be an STP (Shielded Twisted Pair). A 120 Ohm - 1/2 Watt resistor must be installed at each end. These resistors are commonly known as "end of line resistors".
The diagram below shows the architecture of an RS485 link.

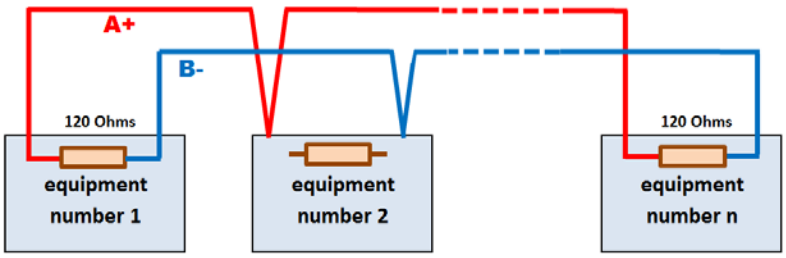


figure 1 - RS485 link architecture

The common software parameters for an RS485 link are as follows:

description	default setting on the **APM303**
communication speed expressed in bauds	**9600**
number of data bits: 7 or 8 bits	**8** (non-modifiable)
number of stop bits: 1 or 2	**1**
type of parity check: none, even, odd	**none**
equipment address: from 1 to 255	**5**

3.2 - Connecting a computer to the RS485 network

This part explains how to connect a PC to the RS485 network. To do so, you need to know whether the computer to be used for the connection is equipped with a serial port.

3.2.1 - Computer equipped with a serial port

The computer serial port has a built-in RS232 serial link. This link is incompatible with the RS485 link. To communicate with the **APM303**, it is necessary to use an RS232/RS485 converter. SDMO offers the AD400E converter to perform this function.
The connection diagram is as follows:

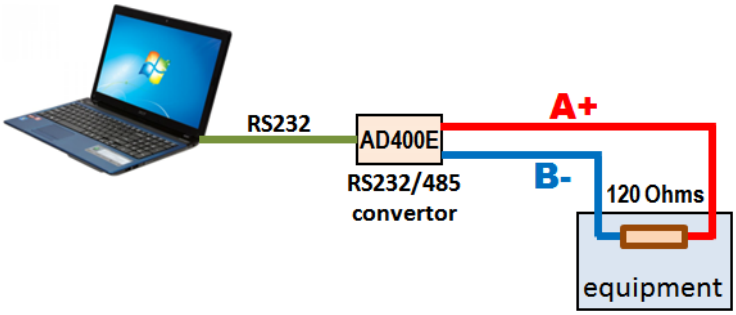


figure 2 - PC connection with serial port

3.2.2 - Computer not equipped with a serial port

This is the most common scenario nowadays. Today's computers practically no longer come with serial ports. The solution recommended by SDMO for connecting a computer not equipped with a serial port to the RS485 network is to use a USB/RS232 converter coupled to the AD400E RS232/RS485 converter.
The connection diagram is as follows:

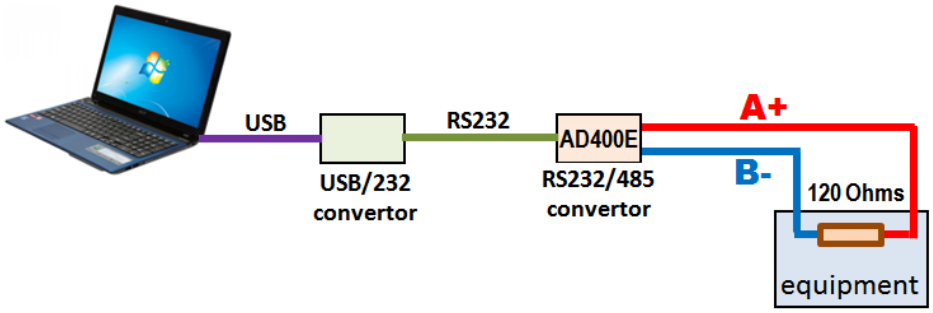


figure 3 - PC connection without serial port

3.3 - Modbus protocol

3.3.1 - General presentation

A protocol is a way of organising the data exchanged on a physical medium (e.g.: RS485, RS232, etc.). For example: the address at the beginning, then the information, and at the end an information transmission check.

The Modbus protocol is at present very widespread in the field of industrial communication, particularly on PLCs. Below are the essential characteristics of the protocol:

- in a network, there is a master and several slaves,
- each slave has a distinct address,
- the master always takes the initiative in communicating,
- the reading and writing operations between equipment are known as "functions".

The schematic representation of a "master/slave" organisation is as follows:

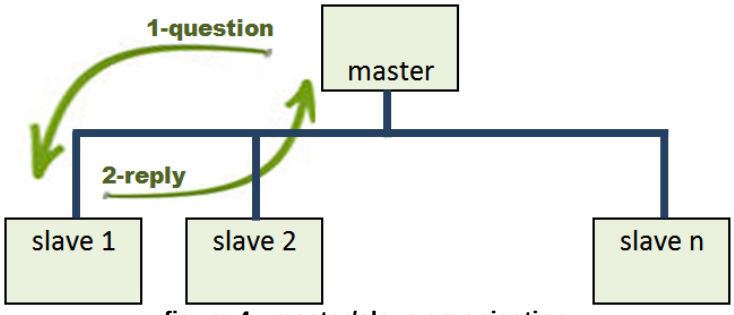


figure 4 - master/slave organisation

3.3.2 - Other protocol characteristics

• ***Slave addressing***
The Modbus protocol starts counting at 1 (table opposite).

slave	decimal
first slave	1
second slave	2
third slave	3
…	…
slave n	n

• ***Processing capacity***
The actual processing capacity depends on the master and slaves on the bus.
Opposite are some figures for the different functions used (see also paragraph 3.3.4).

slave	decimal	hexadecimal
number of slaves	1 to 255	0x001 to 0x0FF
read n bits (function 1)	1 to 2000	0x001 to 0x7D0
read n words (function 3)	1 to 125	0x001 to 0x07D
write n bits (function 15)	1 to 1968	0x001 to 0x7B0
write n words (function 16)	1 to 123	0x001 to 0x07B

3.3.3 - Modbus exchanges

The exchanges are "half-duplex" (first transmitting and then receiving). No slave can send a message without a prior request from the master. Dialogue between slaves is impossible.

The sequence of a dialogue is as follows:

1- The master queries a slave and then waits for its response, both in reading and write mode.
2- The slave responds to the master.

The **APM303** behaves as a slave only. It transmits data over the bus only if requested by the master to do so.
The chosen communication mode is RTU (Remote Terminal Unit) mode: ⇨ the data are in 8 bits.

3.3.4 - Modbus functions and frames

There are 21 Modbus functions. The following functions are implemented in **APM303**:

No.		description (✱)	applicable to …	calculation
01	0x01	read binary outputs (Read Coils)	Internal Bits Or Physical Coils	Bit access
02	0x02	read binary inputs (Read Discrete Inputs)	Physical Discrete Inputs	Bit access
04	0x04	read analogue inputs (Read Input Registers)	Physical Inputs Registers	16 Bits access
05	0x05	write binary output (Write Single Coil)	Internal Bits Or Physical Coils	Bit access

(✱) The terms in brackets are the terms used in the document "Modbus application protocol specification", version V1.1b3, available on *www.modbus.org/*.

All the frames end in a CRC (Cyclic Redundancy Check). The CRC is a mathematical calculation for checking that the frame received is complete. This calculation is generally included in programs using the Modbus protocol. All the frames and data contained are expressed in hexadecimal.

3.3.4.1 - Function 01 (0x01): Read Coils

- The frame sent by the master to request the value of one or more "coils" is as follows:

read request frame	information	slave address	function	coil start address	number of coils to read [1]	CRC
	encoded in …	1 byte	1 byte	2 bytes	2 bytes	2 bytes

[1] The max. number of coils that can be read is 2000 (0x7D0).

- The response sent by the slave is as follows:

response frame	information	slave address	function	number **N** of bytes sent back	values of **N** bytes read	CRC
	encoded in …	1 byte	1 byte	1 byte	**N** x 1 byte	2 bytes

The length of this frame depends on the number **N** of bytes read:
N = whole part of the quotient of the division [**number of inputs/8**], if the remainder is zero
N = whole part of the quotient of the division [**number of inputs/8**], if the remainder is not 0, then **N** = **N**+1
The maximum number **N** will be: 2000/8=250

example: read the status of **10** coils, whole part of [**10/8**] = 1 giving a non-zero remainder, ⇨ **N** = **N**+1 = 2 bytes
read the status of **32** coils, whole part of [**32/8**] = 4 giving a remainder of zero, ⇨ **N** = 4 bytes

- In case of an error in the frame, see paragraph 3.3.4.5.

3.3.4.2 - Function 02 (0x02): read binary inputs (Read Discrete Inputs)

- The frame sent by the master to request the value of one or more binary inputs is as follows:

read request frame	information	slave address	function	input start address	number of inputs to read [1]	CRC
	encoded in …	1 byte	1 byte	2 bytes	2 bytes	2 bytes

[1] The max. number of inputs that can be read is 2000 (0x7D0).

- The response sent by the slave is as follows:

response frame	information	slave address	function	number **N** of bytes sent back	values of **N** bytes read	CRC
	encoded in …	1 byte	1 byte	1 byte	**N** x 1 bytes	2 bytes

The length of this frame depends on the number **N** of bytes sent back:
N = whole part of the quotient of the division [**number of inputs/8**], if the remainder is zero
N = whole part of the quotient of the division [**number of inputs/8**], if the remainder is not 0, then **N** = **N**+1
The maximum number **N** will be: 2000/8=250

example: read the status of **10** inputs, whole part of [**10/8**] = 1 giving a non-zero remainder, ⇨ **N** = **N**+1 = 2 bytes
read the status of **32** inputs, whole part of [**32/8**] = 5 giving a remainder of zero, ⇨ **N** = 4 bytes

- In case of an error in the frame, see paragraph 3.3.4.5.
- For more information on function **02**, refer to the specific examples in paragraphs 5.1.2, 5.1.3 and 5.1.4.

3.3.4.3 - Function 04 (0x04): read multiple registers (Read Input Registers)

- The frame sent by the master to request the value of one or more registers is as follows:

read request frame	information	slave address	function	register start address	number of registers to read [1]	CRC
	encoded in …	1 byte	1 byte	2 bytes	2 bytes	2 bytes

[1] The max. number of registers that can be read is: 125 (0x007D)

- The response sent by the slave is as follows:

response frame	information	slave address	function	number **N** of registers sent back	values of **N** registers read	CRC
	encoded in …	1 byte	1 byte	1 byte	**N** x 2 bytes	2 bytes

The length of the response frame depends on the number **N** of bytes read.

- In case of an error in the frame, see paragraph 3.3.4.5.
- For more information on function **04**, refer to the specific example in paragraph 5.1.1.

3.3.4.4 - Function 05 (0x05): write "a coil" (Write Single Coil)

- The frame sent by the master to change the value of a coil is as follows:

write request frame	information	slave address	function	coil address	value to write in this coil {1}	CRC
	encoded in …	1 byte	1 byte	2 bytes	2 bytes	2 bytes

- The response sent by the slave is as follows:

response frame	information	slave address	function	coil address	value of this coil	CRC
	encoded in …	1 byte	1 byte	2 bytes	2 bytes	2 bytes

If everything is correct, the 2 values are the same, the response is an "echo" of the write request.

- In case of an error in the frame, see paragraph 3.3.4.5.
- For more information on function **05**, refer to the specific example in paragraph 5.2.1.

3.3.4.5 - Error codes and exception codes

In case of error in the frame, the slave sends back an error code and an exception code.

error frame	information	slave address	error code	exception code	CRC
	encoded in …	1 byte	1 byte	1 byte	2 bytes

The error codes are calculated as follows:

error code = function code + 0x80

function	error code
0x01 (01)	81
0x02 (02)	82
0x04 (04)	84
0x05 (05)	85

The table below describes the meaning of the exception codes:

exception code	meaning
01	function code incorrect
02	address incorrect
03	data invalid
4 or 5 or 6	processing error

3.3.4.6 - Length of frames

For functions 01, 02, 04 and 05, implemented in the **APM303** module, the length of the frames is as follows:

nature of frame	function	frame length in bytes
read request	01	8
	02	8
	04	8
write request	05	8
response	01	variable (depending on number of bytes sent back)
	02	variable (depending on number of bytes sent back)
	04	variable (depending on number of bytes sent back)
	05	8

4 - Information available on the APM303 via the RS485 link

This part presents all the information available on the RS485 link. These data are classified according to their type:

- Binary inputs and output (hardware),
- Binary inputs and outputs (logic),
- Remote control,
- Measurements (electrical, mechanical, miscellaneous).

4.1 - Binary inputs and outputs (hardware)

4.1.1 - Binary inputs (hardware)

binary hardware inputs				
information on the frame				
access: read only	note: it is necessary to know the allocation of the APM303 inputs			
query data				
APM303 address: 5 (by default)	Modbus function: 0x02	Modbus address: as indicated below	length of data: as indicated below	
indications available				
address	value	length (bits)	corresponding terminal and allocation	
0x0000	hardware binary input no.1	1	**T04** - emergency stop	
0x0001	hardware binary input no.2	1	**T10** - (see list of "input codes")	
0x0002	hardware binary input no.3	1	**T11** - (see list of "input codes")	
0x0003	hardware binary input no.4	1	**T12** - (see list of "input codes")	
0x0004	hardware binary input no.5	1	**T13** - (see list of "input codes")	
0x0005	hardware binary input no.6	1	**T14** - (see list of "input codes")	
0x0006	hardware binary input no.7	1	**T15** - (see list of "input codes")	

The list of "input codes" (or functions) is available in paragraph 4.2.1.

4.1.2 - Binary outputs (hardware)

binary hardware outputs				
information on the frame				
access: read only	note: it is necessary to know the allocation of the APM303 outputs			
query data				
APM303 address: 5 (by default)	Modbus function: 0x02 (∗)	Modbus address: as indicated below	length of data: as indicated below	
indications available				
address	value	length (bits)	corresponding terminal and allocation	
0x0010	hardware binary output no.1	1	**T05** - starter control	
0x0011	hardware binary output no.2	1	**T06** - fuel solenoid valve control	
0x0012	hardware binary output no.3	1	**T07** - (see list of "output codes") (∗∗)	
0x0013	hardware binary output no.4	1	**T08** - (see list of "output codes") (∗∗)	
0x0014	hardware binary output no.5	1	**T09** - (see list of "output codes") (∗∗)	

(∗) Note that function 02 was chosen for reading the outputs (the frame structure is the same as function 01).
(∗∗) As standard, terminals **T07**, **T08** and **T09** are allocated according to the table below:

output code	function	wiring to …
O08	air preheating control	**T07**
O07	"ready to load" report	**T08**
O15	common shutdown report	**T09**

The complete list of "output codes" (or functions) is available in paragraph 4.2.2.

4.2- Binary inputs and outputs (logic)

4.2.1 - Binary inputs (logic)

This table brings together the functions available for the module **APM303** inputs.

logic binary inputs; functions				
information on the frame				
access: read only	note:			
query data				
APM303 address: 5 (by default)	Modbus function: 0x02	Modbus address: as indicated below	length of data: as indicated below	
indications available				
address	value		length (bits)	function code
0x0060	Emergency Stop		1	-
0x0061	Remote Start/Stop		1	I02
0x0062	Access Lock (APM303 locked)		1	I04
0x0063	GCB feedback		1	I07
0x0064	External Warning 1		1	I10
0x0065	External Warning 2		1	I11
0x0066	External Warning 3		1	I12
0x0067	External Shutdown 1		1	I13
0x0068	External Shutdown 2		1	I14
0x0069	External Shutdown 3		1	I15
0x006A	Low Fuel Level		1	I20
0x006B	Low Oil Pressure		1	I22
0x006C	High Coolant Temperature		1	I24

GCB = Generator Circuit Breaker

4.2.2 - Binary outputs (logic)

This table brings together the statuses, warnings, shutdowns and functions available for the module **APM303** outputs.

binary outputs (logic): functions, statuses, warnings and shutdowns				
information on the frame				
	access: read only	note:		
query data				
	APM303 address: 5 (by default)	Modbus function: 0x02 (∗)	Modbus address: as indicated below	length of data: as indicated below
indications available				
	address	value	length (bits)	code function, status, warning, shutdown
	0x0020	Starter	1	-
	0x0021	Fuel Solenoid	1	-
	0x0022	Stop Solenoid	1	O03
	0x0023	General Alarm	1	O04
	0x0024	GCB Close/Open	1	O05
	0x0025	Ready To Load	1	O07
	0x0026	Preheat	1	O08
	0x0027	Running (generating set stabilised)	1	status
	0x0028	Automatic mode (Auto On/Off)	1	status
	0x0029	Island Operation (ready to load, circuit breaker closed)	1	status
	0x002A	Common Warning	1	O14
	0x002B	Maintenance Required	1	warning
	0x002C	Low Battery	1	warning
	0x002D	Low Fuel Level	1	warning or shutdown (∗∗)
	0x002E	External Warning 1	1	warning
	0x002F	External Warning 2	1	warning
	0x0030	External Warning 3	1	warning
	0x0031	Generator CCW Rotation	1	warning
	0x0032	Battery Flat	1	warning
	0x0033	Common Shutdown	1	O15
	0x0034	Emergency Stop Active	1	shutdown
	0x0035	Overspeed	1	shutdown
	0x0036	Underspeed	1	shutdown
	0x0037	Low Oil Pressure	1	shutdown
	0x0038	High Coolant Temperature	1	shutdown
	0x0039	External Shutdown 1	1	shutdown
	0x003A	External Shutdown 2	1	shutdown
	0x003B	External Shutdown 3	1	shutdown
	0x003C	GCB Fail (circuit breaker position inconsistent)	1	shutdown
	0x003D	Generator V> (max. generator voltage)	1	shutdown
	0x003E	Generator V< (min. generator voltage)	1	shutdown
	0x003F	Generator Hz> (max. generator frequency)	1	shutdown
	0x0040	Generator Hz< (min. generator frequency)	1	shutdown
	0x0041	Start Fail	1	shutdown
	0x0042	Stop Fail	1	shutdown
	0x0043	Generator A>> (generator short-circuit)	1	shutdown
	0x0044	Generator kW> (generating set overload)	1	warning
	0x0045	Choke	1	O10
	0x0046	Glow Plugs	1	O11
	0x0047	Valve Extinguisher	1	O13

(∗) Note that function 02 was chosen for reading the functions, statuses, warnings and shutdowns (the frame structure is the same as function 01).
GCB = Generator Circuit Breaker
(∗∗) depending on configuration

4.3 - Remote control

This table brings together the 3 functions for remote controlling the operating mode of the **APM303** module.
It is assumed that the operating modes are coils to which a datum is written.

remote control				
information on the frame				
	access: write	note:		
query data				
	APM303 address: 5 (by default)	Modbus function: 0x05	Modbus address: as indicated below	length of data: as indicated below
indications available				
	address	value	length (bits)	
	0x0000	Remote button START	1	-
	0x0001	Remote button STOP	1	-
	0x0002	Remote AUTO-MODE	1	-

4.4 - Electrical and mechanical measurements

This table brings together all the analogue measurements available on the **APM303** module, and some miscellaneous information, such as the hardware version number.

electrical and mechanical measurements						
information on these indications						
access: read only	note: the length of the data to read must be taken into account. The values sent back are signed or unsigned					
query data						
APM303 address: 5 (by default)	Modbus function: 0x04	Modbus address: as indicated below	length of data: as indicated below			
indications available						
address	data	type	length (bits)	value	unit	
0x0000	Gen V L1-N (live 1/neutral voltage)	unsigned	16	1	V	
0x0001	Gen V L2-N (live 2/neutral voltage)	unsigned	16	1	V	
0x0002	Gen V L3-N (live 3/neutral voltage)	unsigned	16	1	V	
0x0003	Gen V L1-L2 (live 1/live 2 voltage)	unsigned	16	1	V	
0x0004	Gen V L2-L3 (live 2/live 3 voltage)	unsigned	16	1	V	
0x0005	Gen V L1-L3 (live 1/live 3 voltage)	unsigned	16	1	V	
0x0006	Gen A L1 (live 1 current)	unsigned	16	1, 1/10 **(1)**	A	
0x0007	Gen A L2 (live 2 current)	unsigned	16	1, 1/10 **(1)**	A	
0x0008	Gen A L3 (live 3 current)	unsigned	16	1, 1/10 **(1)**	A	
0x0009	Gen kW Total (total active power)	integer	16	1, 1/10 **(1)**	kW	
0x000A	Gen kVA Total (total effective power)	Integer	16	1, 1/10 **(1)**	kVA	
0x000B	Gen PF Total (power factor)	integer	16	1/100	-	
0x000C	Gen kW L1 (live 1 active power)	integer	16	1, 1/10 **(1)**	kW	
0x000D	Gen kW L2 (live 2 active power)	integer	16	1, 1/10 **(1)**	kW	
0x000E	Gen kW L3 (live 3 active power)	integer	16	1, 1/10 **(1)**	kW	
0x000F	Gen kW L1 (live 1 effective power)	integer	16	1, 1/10 **(1)**	kVA	
0x0010	Gen kW L2 (live 2 effective power)	integer	16	1, 1/10 **(1)**	kVA	
0x0011	Gen kVA L3 (live 3 effective power)	integer	16	1, 1/10 **(1)**	kVA	
0x0012	Gen PF L1 (live 1 power factor)	signed	8	1/100	-	
0x0013	Gen PF L2 (live 2 power factor)	signed	8	1/100	-	
0x0014	Gen PF L3 (live 3 power factor)	signed	8	1/100	-	
0x0015	Load character	unsigned	8	R/L/C **(2)**	**-**	
0x0016	RPM (engine speed)	unsigned	16	1	RPM	
0x0017	Gen Freq	unsigned	16	1/10	Hz	
0x0018	Power reading precision **(3)**	unsigned	16	-	-	
0x0019	U bat (battery voltage)	integer	16	1/10	V	
0x001A	BIN **(4)**	unsigned	16	-	-	
0x001B	BOUT **(5)**	unsigned	16	-	-	
0x001C	Oil pressure	integer	16	1, 1/10 **(1)**	**(6)**	
0x001D	Coolant temperature	integer	16	1	**(7)**	
0x001E	Fuel level	integer	16	1	%	
0x001F	Unit system (metric/US)		8	(✱)	-	
0x0020	D plus (charging alternator)	integer	16	1/10	V	
0x0021	kWh (active energy)	integer	32	Hi	kWh	
0x0022				Lo	kWh	
0x0023	Maintenance timer	unsigned	16	1	H	
0x0024	Start counter	unsigned	16	1	-	
0x0025	GensetSerialNumber	unsigned	32	Hi	-	
0x0026				Lo	-	
0x0027	SW version **(8)**	unsigned	16	-	-	
0x0028	SW patch version **(9)**	unsigned	16	-	-	

(✱) 0 = metric system, 1=US system
for **(1) (2) (3) (4) (5) (6) (7) (8) (9)**: see explanations on next page

(1)	If APM303 is "decimal" configured via the parameter “Power reading precision”, then some values will be displayed with a decimal point (*example*: Gen A L1= 10.7A) ⇨ equivalent of a multiplier on certain analogue values: 1 or 0.1
(2)	The characters R, L, C are sent in Ascii: MSB->0, LSB->CHAR, empty character: space, 0x20: if no power measured
(3)	If value 0 then no decimal, if value 1 then 1 decimal
(4)	Physical status of a binary input
(5)	Physical status of a binary output
(6)	Depending on configuration (bar or psi)
(7)	Depending on configuration (°C or °F)
(8)	M (major version) x 100 + N (minor version) (*example*: 101 represents version 1.1) (✱)
(9)	P (patch) x 1000 B (build) (*example*: 2005 represents patch 2.5) (✱)

(✱) With the 2 examples above, a complete software version would be: 1.1.2.5

5 - Example applications of RS485

This paragraph sets out specific examples of applications of Modbus communication with the **APM303**.
For all these examples, the **AMP303** address should be checked on the "Basic settings" screen, or in the "Setpoints" menu, "Modbus parameters" tab. Let's assume an **APM303** at address **05** (standard setting).

5.1 - Information reading

5.1.1 - Battery voltage reading

The battery voltage information can be found in paragraph 4.4. The characteristics to remember are as follows:
- Modbus function: **0x04** (**Read Input Registers**),
- address to query: **0x0019** (25 in decimal),
- length of datum to query: 16 bits or 2 bytes,
- the battery voltage unit of measurement is tenths of a Volt.

These characteristics are used to put together the request frame:

read request frame	information	slave address	function	address of register to read	number of registers to read	CRC (∗)
	encoded in …	1 byte	1 byte	2 bytes	2 bytes	2 bytes
	content	**05**	**04**	**00 19** (MSB LSB)	**00 01** (∗∗) (MSB LSB)	**xx xx** (LSB MSB)

(∗∗) In the example, there is only one register to read, i.e. **01** in hexadecimal

The **APM303** sends back the response frame below (example):

response frame	information	slave address	function	number **N** x 2 of bytes sent back	values of **N** registers read	CRC (∗)
	encoded in …	1 byte	1 byte	1 byte	**N** x 2 bytes	2 bytes
	value	**05**	**04**	**02** (∗∗∗)	**00 EF** (MSB LSB)	**xx xx** (LSB MSB)

(∗) The value of CRC needs to be calculated (*note*: in most systems, this value is returned automatically).
(∗∗∗) **N** = 1 register to be read x 2 = 2, i.e. value 02 in hexadecimal
The input registers are sent back in 2 bytes, since the size of the registers is at least 1 word (2 words for many analogue data) (see table in paragraph 4.4).
example: at address 0x0003: Gen V L1-L2 (voltage between live 1 and 2) is encoded in 1 word or 2 bytes, since with 1 byte, it is impossible to count above 255 (**FF**). With 2 bytes, it is possible to count from 0 to 65535 (**FF FF**) (*example*: 410 Volts between live 1 and live 2 will be denoted **01 9A**).

MSB = Most Significant Byte
LSB = Least Significant Byte

The information to check is as follows:
- same slave address as the request frame,
- same function as the request frame,
- number of bytes sent back = number **N** x 2.

After these checks, the battery voltage is read by converting the hexadecimal value into a decimal value.

hexadecimal value	decimal value	multiplier (see section 4.4)	battery voltage
00 EF	**239**	**1/10**	**23.9 Volts**

All the other information from paragraph 4.4 must be read in the same way. Make sure to respect the length of each datum.

5.1.2 - Reading “Ready To Load” status

The "Ready To Load" information can be found in paragraph 4.2.2. The characteristics to select are as follows:

- Modbus function: **0x02** (**Read Discrete Inputs**),
- address to query: **0x0025** (37 in decimal),
- length of datum to query: 1 bit.

These characteristics are used to put together the request frame:

read request frame	information	slave address	function	start address to read	number of inputs to read	CRC (∗)
	encoded in …	1 byte	1 byte	2 bytes	2 bytes	2 bytes
	content	**05**	**02**	**00 25** (MSB LSB)	**00 01** (∗∗) (MSB LSB)	**xx xx** (LSB MSB)

(∗∗) In the example, there is only one input to read.

APM303 sends back the response frame below (example):

response frame	information	slave address	function	number **N** of bytes sent back	values of **N** inputs read	CRC (∗)
	encoded in …	1 byte	1 byte	1 byte	**N** x 1 byte	2 bytes
	value	**05**	**02**	**01** (∗∗∗)	**01** (∗∗∗∗)	**xx xx** (LSB MSB)

(∗) The value of CRC needs to be calculated. In most systems, this value is returned automatically.
(∗∗∗) **N** = 1 input/8=0 + remainder not 0 ⇨ **N** =0+1=1 byte sent back
(∗∗∗∗) In the example, there is only one input. The result in binary of **01** (value in hexadecimal) is: **0000 0001**.
This result shows that the queried input is at status **1** ("Ready To Load" is activated). The other bits in the byte containing the input status are 0.

MSB = Most Significant Byte
LSB = Least Significant Byte

The information to check is as follows:

- same slave address as the request frame,
- same function as the request frame.
- number of bytes sent back ⇨ see calculation for **N** (∗∗∗).

5.1.3 - Reading the "Common Shutdown" status

The "Common Shutdown" information can be found in paragraph 4.2.2. The characteristics to remember are as follows:

- Modbus function: **0x02** (**Read Discrete Inputs**),
- address to query: **0x0033** (51 in decimal),
- length of datum to query: 1 bit.

These characteristics are used to put together the request frame:

read request frame	information	slave address	function	start address to read	number of inputs to read	CRC (✱)
	encoded in …	1 byte	1 byte	2 bytes	2 bytes	2 bytes
	content	**05**	**02**	**00 33** (MSB LSB)	**00 01** (✱✱) (MSB LSB)	**xx xx** (LSB MSB)

(✱✱) In the example, there is only one input to read.

APM303 sends back the response frame below (example):

response frame	information	slave address	function	number **N** of bytes sent back	values of **N** inputs read	CRC (✱)
	encoded in …	1 byte	1 byte	1 byte	**N** x 1 byte	2 bytes
	value	**05**	**02**	**01** (✱✱✱)	**01** (✱✱✱✱)	**xx xx** (LSB MSB)

(✱) The value of CRC needs to be calculated. In most systems, this value is returned automatically.
(✱✱✱) 1 byte sent back, since 1/8=0 and remainder not zero, so **N**=N+1=1.
(✱✱✱✱) In the example, there is only one input. The result in binary of **01** (value in hexadecimal) is: **0000 0001**.
This result shows that the queried input is at status **1** ("Common Shutdown" is activated). The other bits in the byte are at 0.

MSB = Most Significant Byte
LSB = Least Significant Byte

The information to check is as follows:

- same slave address as the request frame,
- same function as the request frame,
- number of bytes sent back ⇨ see calculation for N (✱✱✱).

5.1.4 - Reading the status of x contiguous inputs

It is also possible to read the status of "Common Shutdown" by querying **x** contiguous inputs. Let's assume that the generating set has been shut down for an oil pressure problem, and that the **APM303** was set to AUTO mode before the fault shutdown. The "Common Shutdown" information can be found in paragraph 4.2.2.

The characteristics to remember are as follows:

- Modbus function: **0x02** (**Read Discrete Inputs**),
- addresses to query: **0x0020** (32 in decimal) to **0x0037** (55 in decimal),
- length of datum to query: 1 bit.

These characteristics are used to put together the request frame:

read request frame	information	slave address	function	start address to read	Number of inputs to read	CRC
	encoded in …	1 byte	1 byte	2 bytes	2 bytes	2 bytes
	content	**05**	**02**	**00 20**	**00 18** (✱)	**xx xx**

(✱) 24 inputs to read, i.e. **00 18** in hexadecimal (*reminder*: it is possible to read up to 2000 inputs, i.e. **07 D0**).

APM303 sends back the response frame below (example):

response frame	information	address	function	number **N** of bytes sent back	values of **N** inputs read			CRC
	encoded in …	1 byte	1 byte	1 byte	**N** x 1 byte			2 bytes
	value	**05**	**02**	**03** (✱✱)	**08**	**A1**	**88**	**xx xx**
					byte 1	byte 2	byte 3	
					(✱✱✱)			

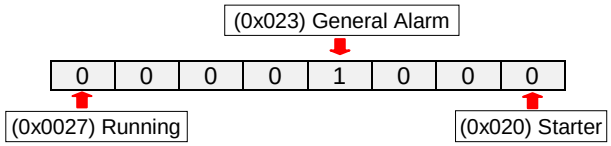
(✱✱) In the example, there are 24 inputs to query: 24/8=3 (the remainder is zero), i.e. 3 bytes sent back.

(✱✱✱) Each byte contains the status of 8 “Real Discrete Inputs”:
- byte 1 contains the values at the addresses 0x0020 to 0x0027,
- byte 2 contains the values at the addresses 0x0028 to 0x002F,
- byte 3 contains the values at the addresses 0x0030 to 0x0037.

The details of each byte are as follows:

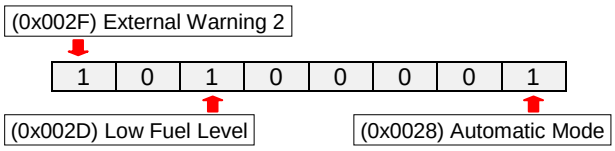
byte 1:
- the bit at address 0x0020 is on the right,
- the bit at address 0x0027 is on the left.

Since "Common Shutdown" is activated, "General Alarm" is also activated, the other bits are 0, which in hexadecimal gives: **08** (00001000 in binary)



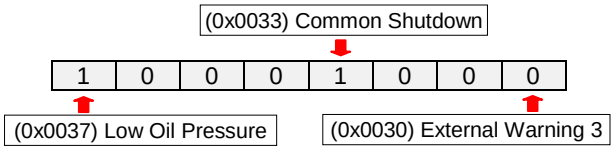
byte 2:
- the bit at address 0x0028 is on the right,
- the bit at address 0x002F is on the left.

"Automatic Mode" is activated and we assume that "Low Fuel Level" and "External Warning 2" could be activated, which gives in hexadecimal: **A1** (10100001 in binary)



byte 3:
- the bit at address 0x0030 is on the right,
- the bit at address 0x0037 is on the left.

"Low Oil Pressure" is at 1, "Common Shutdown" is at 1, which gives in hexadecimal: **88** (10001000 in binary)



5.2 - Information writing

5.2.1 - Sending a command to the generating set

Let us assume that the AUTO button has been pressed remotely. The AUTO button information can be found in paragraph 4.3. The characteristics to remember are as follows:
- Modbus function: **0x05** (**Write Single Coil**),
- address to query: **0x0002** (2 in decimal),
- length of data to query: 1 bit.

These characteristics are used to put together the request frame:

write request frame	information	slave address	function	coil address	value to write in this coil	CRC (✱)
	encoded in …	1 byte	1 byte	2 bytes	2 bytes	2 bytes
	content	**05**	**05**	**00 02** (MSB LSB)	**FF 00** (✱✱) (MSB LSB)	**xx xx** (LSB MSB)

APM303 sends back the following response frame:

response frame	information	slave address	function	coil address	value of this coil	CRC (✱)
	encoded in …	1 byte	1 byte	2 bytes	2 bytes	2 bytes
	content	**05**	**05**	**00 02** (MSB LSB)	**FF 00** (✱✱) (MSB LSB)	**xx xx** (LSB MSB)

(✱) The value of CRC needs to be calculated. In most systems, this value is returned automatically.

(✱✱) **00 00**: AUTO button = OFF **FF 00**: AUTO button = ON
Any other value does not affect the coil status.

MSB = Most Significant Byte
LSB = Least Significant Byte

The information to check is as follows:
- same slave address as the write request frame,
- same function as the write request frame,
- same result for the coil "address" and "value" boxes,

5.3 - Frame capture with ModBus Doctor

Modbus Doctor is a freeware available on the Internet. It is used for monitoring frames sent and received after entering a query.
After configuring the parameters required for connection, the following results are obtained:

■ **Capture of two frames with function 04 Battery voltage value**

frame one:
-read request in 8 bytes
-queried address: 00 19 (bytes 2 & 3)
-CRC = E1 89 (bytes 6 & 7)

frame two:
-response in 7 bytes
-value sent back: 00 EF (bytes 3 & 4)
-CRC = 09 7C (bytes 5 & 6)

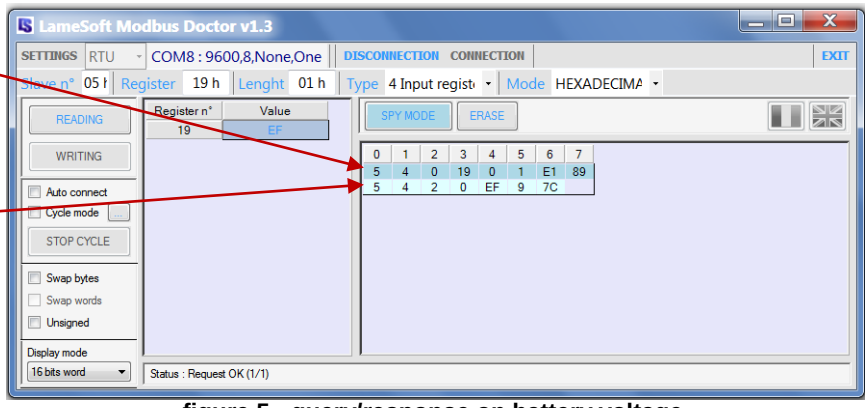


figure 5 - query/response on battery voltage

■ **Capture of two frames with function 02 "Common Shutdown" status**

frame one:
-read request in 8 bytes
-queried address: 00 33 (bytes 2 & 3)
-CRC = 48 41 (bytes 6 & 7)

frame two:
-response in 6 bytes
-value sent back: 01 (byte 3)
-CRC = 61 78 (bytes 4 & 5)

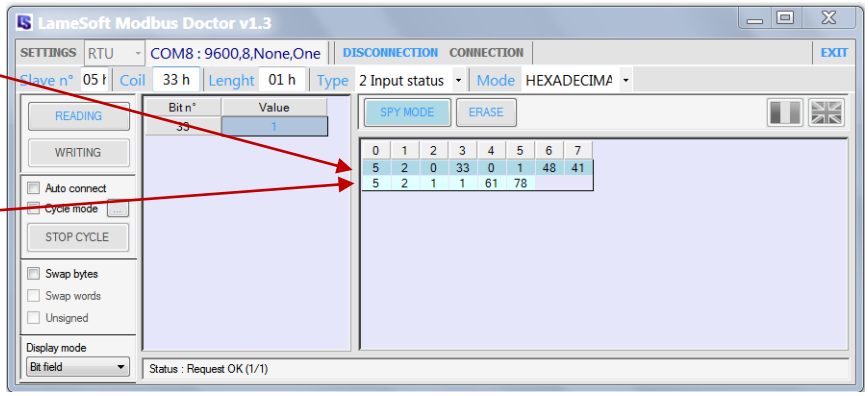


figure 6 - query/response on "Common Shutdown" status

■ **Capture of two frames with function 02 Status of output T09 configured at O15**

frame one:
-read request in 8 bytes
-queried address: 00 14 (bytes 2 & 3)
-CRC = F8 4A (bytes 6 & 7)

frame two:
-response in 6 bytes
-value sent back: 01 (byte 3)
-CRC = 61 78 (bytes 4 & 5)

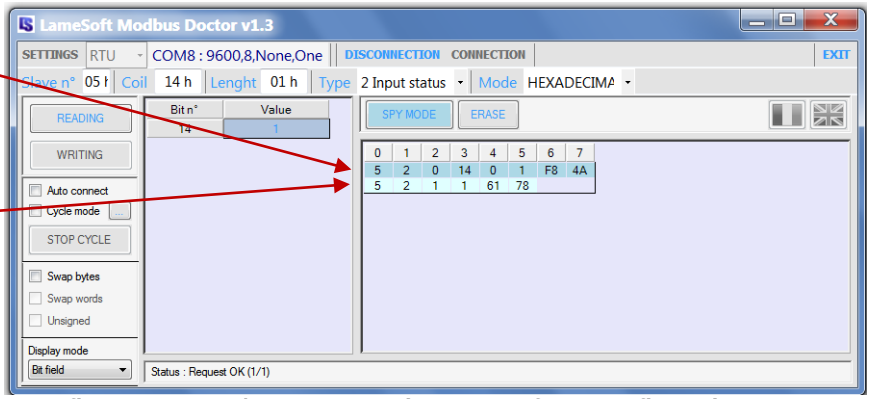


figure 7 - query/response on the status of T09 configured at O15

PERSONAL NOTES