KMB systems, s.r.o.
Dr. M. Horákové 559, 460 06 Liberec 7, Czech Republic
tel. +420 485 130 314, fax +420 482 736 896
email : kmb@kmb.cz, internet : www.kmb.cz

AFR 111, AFR 131
Smart Load for Reduction of Ferroresonance
Communication Protocol Manual

Firmware v. 8.2 / 23. 11. 2018

rev1.1

03/2015

1. Communication Between the Device and the Host System

The AFR 1xx Smart Load can be equipped either with “remote” communication interfaces : RS-485.

The remote serial communication ports support the Modbus-RTU protocols to provide all information to the operator's PC.

1.1 The Modbus Protocol Description

The instruments implement a Modbus-RTU interface. For more details about various Modbus implementations see http://www.modbus.org/specs.php .

Instruments with the RS-485 serial line can be configured to use the Modbus-RTU protocol . For this option the address, baud rate and parity bit are specified (see the respective part of user manual for configuration details). A gap between bytes corresponding to maximum 1.5 characters (bytes) is allowed while receiving a command or transmitting a reply.

1.1.1 Supported Functions

- 3 (0x03) ... read holding registers
- 4 (0x04) ... read input registers
- 16 (0x10) ... write multiple registers

The “broadcast” mode is not supported.

1.1.2 Modbus Quantity Encoding

Access to data structure components is provided using read/write from/to relevant registers as shown in the chart in the following subsections. Modbus protocol is based on variable mappings into 16 bit registers. Single-byte quantities are stored in such a register in the format of 0x00nn where nn is a single-byte parameter. For multibyte quantities the byte ordering is a big endian. 32-bit and 64-bit integers and floats are ordered in consequent 16-bit registers from MSB to LSB serially. Floats are encoded using the IEEE-754 float number format.

Structures which hold the instrument setting information are stored in an array of ‘holding’ registers, that can be written or read. The currently measured data and the instrument status can be read as the contents of the ‘input’ registers, that er read only. Each structure component is stored within the array of registers using the base addresses and a given offset.

Each value is encoded in the following way:

1. If not stated otherwise, each register is encoded in the same way as the corresponding variable in the respective message of the KMB Long protocol (Section 3).
2. Actual data values of float type (voltage, current, powers, etc.) hold the value of the respective quantity, no recalculation is needed.

ANSI C and .NET functions (sample code) for time and value conversions can be provided upon request.

1.1.3 Data Structure Register Maps

IR = input register, 16-bit, read only

HR = holding register, 16-bit, read and write

data structure	base address	type
Instrument Identification	0x0200 (512 dec)	IR
Communication Setup	0x0800 (2048 dec)	IR, HR
Actual Data	0x1000 (4096 dec)	IR
Data logger	0x2000 (8192 dec)	IR, HR

Generally, IR and HR have separated address spaces. For simplicity those are now taken together so it is possible to map the HR into the IR space and read them also as the IR with function 0x04.

1.1.4 Data Mapping in 16-bit Registers

Used formats :

int8 … 8-bit integer (signed)
uint8 … unsigned 8-bit integer
int16 … 16-bit integer (signed)
uint16 … unsigned 16-bit integer
int32 … 32-bit integer (signed)
uint32 … unsigned 32-bit integer
int64 … 64-bit integer (signed)
uint64 … unsigned 64-bit integer
float … 32-bit IEEE754-format float

Instrument Identification

IR, base address 0x0200 (512 dec)

register offset (dec)	format	value
0 (0x0000)	uint16	serial number
1 (0x0001)	uint16	instrument type code ●
2 (0x0002)	uint16	props-type code (0x3000)
3 (0x0003)	uint16	firmware version *100 (822 = 8.22)
4 (0x0004)	uint16	hardware version *100 (302 = 3.02)
5 (0x0005)	uint16	bootloader version (0xffff)
6 ÷ 7 (0x0006)	uint32	Reserve
8 (0x0008)	uint8	Number of calibrations
8,5	uint8	Input voltage offset (128±2)
9 (0x0009)	uint16	Temperature sensor offset
10 (0x000a)	uint16	Internal temperature sensor offset

11 (0x000b)	uint16	Resistance of power resistor (Ohms)
12 (0x000c)	uint16	Resistance of power resistor 2 (Ohms)
13 (0x000d)	uint16	Voltage tranfer ratio *10 (multipied 10)

PropsType : 0x3000 for AFR1xx

- InstrumentType :

Bits 15-7 : zeros (reserve)
Bit 6 : X=1(one phase AFR1xx)...0, X=3(open delta three phase AFR3xx)...1
Bit 5 : Y=0(no datalogger)...0, Y=1(datalogger)...1
Bit 4 : Z=N(powered form protected transformer)...0, Z=S(auxiliary supply)...1

Bits 3-0 nominal voltage :
Bit 3 : denominator=1...0, denominator=V3...1
Bity 2-0 nominator of voltage:
100V ... 000
110V ... 001
120V ... 010
220V ... 011
230V ... 100
30V ... 111
???V ... other for future

Communication Setup

IR, HR, base address 0x0800 (2048 dec)

register offset (dec)	format	value
0 (0x0000)	uint16	-
1 (0x0001)	uint8	communication slave address (RS-485)
2 (0x0002)	uint8	communication rate (RS-485) 0 = 4800 Bd 1 = 9600 Bd 2 = 19200 Bd 3 = 38400 Bd 4 = 57600 Bd 5 = 115200 Bd 6 = 230400 Bd
3 (0x0003)	uint8	communication protocol (RS-485) 0 = Modbus / KMB Short*, 8 bit, no parity, 1 stop bit 1 = Modbus RTU, 8 bit, no parity, 2 stop bits 2 = Modbus RTU, 8 bit, even parity, 1 stop bit 3 = Modbus RTU, 8 bit, odd parity, 1 stop bit

* KMB Short - identification only supported

Actual Data

IR, base address 0x1000 (4096 dec)

register offset (dec)	format	value	unit
0 ÷ 255 (0x0000÷00ff)	uint8, uint8	Waveform buffer (row of 512 ADC samples, substract offset 128) ●	Raw data
256 (0x0100)		Reserved	
257 (0x0101)	uint8	Previous state of dataloger* (low byte)	
	uint8	Actual state of datalogger* (high byte)	
258 (0x0102)	uint16	Actual temperature †	Raw data
259 (0x0103)	uint8	Last reset source of microcontroler (see datalogger entry) (low byte)	
	uint8	Actual voltage amplitude *0.004297*Voltage tranfer ratio (see Instrument identification entry)	V
260 (0x0104)	uint32	Last state of dataloger duration	s
262 (0x0106)	uint8	Actual connected resistance -maximal value during state ↕ (low byte)	Raw data
	uint8	Actual connected resistance ‡ (high byte)	Raw data
263 (0x0107)	uint16	Fan state (bit 1 = on, bit 0 = off)	

- Sampling is trggered by reading Communication Setup (0x800)

* 0x00 ;Start (+ info. reset source)
0x20 ;Normal votage
0x40 ;AFR area activation (+ info. max R stage)
0x60 ;AFR DCarea activation (+ info. max R stage)
0x80 ;over temperature (+ info. teperature)
0xa0 ;Low voltage
0xe0 ;OFF

†
```
R0 = 10e3;
beta = 4300/1.0;
T0 = 25 + 273.15;
R39 = 2000;
Vref = 65536;
U = data16; %temperature data
R = R39/(Vref/U-1);
T = T0.*beta./(beta + T0.*(log(R) - log(R0))) - 273.15;
%in celsius degree
```

‡ R = (31 – uint8)*Resistance of power resistor [Ohm]

↕ R = (uint8)*Resistance of power resistor [Ohm]

Dataloger data

IR, HR, base address 0x2000 (8192 dec)

register offset (dec)	format	value	unit
0	uint16	Poiner at top of record	
1	uint4	Pointer Checksum (4bit nibles xor) – 4 high bits (0xf0)	
	uint4	Pointer errors – 4 low bits (0x0f)	
	uint 8	How many times was pointer reseted	
4 ÷ ram end		Dataloger entries	

Dataloger entry (8 bytes)

Entry offset (dec)	format	value	unit
0	uint8	Type of entry (bits 5-7), associated value*	
0,5 ÷ 1	uint24	State duration	s
2	uint16	State duration (10800 fraction of second)	
3	uint8	Maximal temperature during state	Raw data
3,5	uint8	Check summ (8bit one's complemen sum of previous 7 bytes + preseed value 0xaa)	

*Type of entry and assosiated value

Type of entry bits 7 ÷ 5	Value bits 4 ÷ 0	Meaning	unit
0 (0x0..)		Microcontroler start	
	1	Power-on reset (normal start)	
	2	External reset (after programming)	
	4	Brown-out reset (normal start)	
	8	Watchdog reset (abnormal situation)	
1 (0x2..)	0 ÷ 31	Normal system votage *6.24	V
2 (0x4..) 3 (0x6..)		Activation	
	0 ÷ 31	Minimal resistance (means maximal reached R stage)	Raw data
5 (0xa..)		Low system voltage	
	0 ÷ 31	Voltage level	Raw data
7 (0xe..)		Off – low mikrocontroler supply voltage	
	0 ÷ 29	Mikrokontroler voltage level	Raw data
7 (0xfe)	30	Top of record (check consistency with Pointer at top)	

1.1.5 Modbus Communication Examples

1.1.5.1 Reading Device Identification Example

Request: **01 04 02 00 00 06 71 B0**

01 … device address

04 ... reading input registers (IR)

02 00 ... start address (0x200=512 dec, no. of register - 1)

00 06 ... register count

71 B0 … CRC-16

Answer: **01 04 0C 00 15 11 04 00 40 0B D6 00 00 06 50 B8 DA**

01 ... device address

04 ... reading input registers (IR)

0C ... data byte count (0x0C=12 dec 16-bit registers)

00 15 … serial number 21

11 04 ... instrument type code

00 40 ... props-type code

0B D6 ... firmware version

00 00 ... hardware version

06 50 ... bootloader version

B8 DA CRC-16

1.1.5.2 Reading Configurable Settings Example\

Request: **01 03 07 00 00 09 31 78**

Answer: **01 04 12 FF FF 00 01 A3 28 80 05 00 05 43 66 00 00 43 8E DB 6E F4 28**

1.1.5.3 2.5.3 Write Into (Modify) the Configurable Settings

Request: **01 10 07 00 00 09 12 FF FF FF FF 00 01 00 01 00 05 43 66 00 00 42 C8 00 00 54 11**

Answer: **01 10 07 00 00 09 33 31**