COMPALARM A

Annunciatore d'allarme

PROTOCOLLO DI COMUNICAZIONE MODBUS

PROTOCOLLO MODBUS

La scheda d'allarme AC65-485 supporta il protocollo di comunicazione Modbus RTU sulla porta seriale RS-485.

Quando si utilizza il protocollo Modbus RTU, la struttura del messaggio di comunicazione è così costituita:

Start frame	Indirizzo (8 bit)	Funzione (8 bit)	Dati (N x 8 bit)	CRC (16bit)	End frame

- Il campo Indirizzo contiene l'indirizzo dello strumento slave cui il messaggio viene inviato.
- Il campo Funzione contiene il codice della funzione che deve essere eseguita dallo slave.
- Il campo Dati contiene i dati inviati allo slave o quelli inviati dallo slave come risposta ad una domanda.
- Il campo CRC consente sia al master che allo slave di verificare se ci sono errori di trasmissione.

FUNZIONI MODBUS

Le funzioni disponibili sono:

03H = Read input register	Consente la lettura delle misure disponibili nell'AC65-485
10H = Preset multiple register	Permette la scrittura di più parametri
11H = Report slave ID	Permette di leggere informazioni relative all'apparecchio

Tempo di risposta AC-65:

- Caso tipico: 150ms
- Caso peggiore: 300ms

COMPALARM A

Alarm annunciator

MODBUS COMMUNICATION PROTOCOL

MODBUS PROTOCOL

The AC65-485 alarm board supports the communication protocol Modbus RTU on the RS-485 serial port.

If one selects the Modbus RTU protocol, the structure communication message has the following structure:

- The Address field holds the serial address of the slave destination device.
- The Function field holds the code of the function that must be executed by the slave.
- The Data field contains data sent to the salve or data received from the slave in response to a query.
- The CRC field allows the master and slave devices to check the message integrity.

MODBUS FUNCTIONS

The available functions are:

03H = Read input register	Allows to read the AC65-485 measures
10H = Preset multiple register	Allows writing several parameters
11H = Report slave ID	Allows to read information about the device

AC-65 response time:

- Typical case: 150ms
- Worst case: 300 ms

CALCOLO DEL CRC (CHECKSUM per RTU)

Esempio di calcolo:

```
static unsigned char auchCRCHi [ ] = {
0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,
0x40,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,  0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,
0x80,  0x41,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,  0x00,  0xC1,  0x81,  0x40,  0x01,
0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,  0x01,  0xC0,  0x80,  0x41,
0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,  0x00,  0xC1,  0x81,
0x40,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,  0x01,  0xC0,
0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,  0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,  0x01,
0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,
0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,
0x40,  0x01,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,
0x80,  0x41,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,  0x00,  0xC1,  0x81,  0x40,  0x01,
0xC0,  0x80,  0x41,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,
0x00,  0xC1,  0x81,  0x40,  0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,
0x40,  0x01,  0xC0,  0x80,  0x41,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1  0x81,  0x40,  0x01,  0xC0,
0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,  0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,  0x01,
0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,  0x40,  0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,
0x00,  0xC1,  0x81,  0x40,  0x01,  0xC0,  0x80,  0x41,  0x01,  0xC0,  0x80,  0x41,  0x00,  0xC1,  0x81,
0x40
} ;

static unsigned char auchCRCLo [ ] = {
0x00,  0xC0,  0xC1,  0x01,  0xC3,  0x03,  0x02,  0xC2,  0xC6,  0x06,  0x07,  0xC7,  0x05,  0xC5,  0xC4,
0x04,  0xCC,  0x0C,  0x0D,  0xCD,  0x0F,  0xCF,  0xCE,  0x0E,  0x0A,  0xCA,  0xCB,  0x0B,  0xC9,  0x09,
0x08,  0xC8,  0xD8,  0x18,  0x19,  0xD9,  0x1B,  0xDB,  0xDA,  0x1A,  0x1E,  0xDE,  0xDF,  0x1F,  0xDD,
0x1D,  0x1C,  0xDC, 0x14,  0xD4,  0xD5,  0x15,  0xD7,  0x17,  0x16,  0xD6,  0xD2,  0x12,  0x13,  0xD3,
0x11,  0xD1,  0xD0,  0x10,  0xF0,  0x30,  0x31,  0xF1,  0x33,  0xF3,  0xF2,  0x32,  0x36,  0xF6,  0xF7,
0x37,  0xF5,  0x35,  0x34,  0xF4,  0x3C,  0xFC,  0xFD,  0x3D,  0xFF,  0x3F,  0x3E,  0xFE,  0xFA,  0x3A,
0x3B,  0xFB,  0x39,  0xF9,  0xF8,  0x38,  0x28,  0xE8,  0xE9,  0x29,  0xEB,  0x2B,  0x2A  0xEA,  0xEE,
0x2E,  0x2F,  0xEF,  0x2D,  0xED,  0xEC,  0X2C,  0xE4,  0x24,  0x25,  0xE5,  0x27,  0xE7,  0xE6,  0x26,
0x22,  0xE2,  0xE3,  0x23,  0xE1,  0x21,  0x20,  0xE0,  0xA0,  0x60,  0x61,  0xA1,  0x63,  0xA3,  0xA2,
0x62,  0x66,  0xA6,  0xA7,  0x67,  0xA5,  0x65,  0x64,  0xA4,  0x6C,  0xAC,  0xAD,  0x6D,  0xAF,  0x6F,
0x6E,  0xAE,  0xAA,  0x6A,  0x6B,  0xAB,  0x69,  0xA9,  0xA8,  0x68,  0x78,  0xB8,  0xB9,  0x79,  0xBB,
0x7B,  0x7A,  0xBA,  0xBE,  0x7E,  0x7F,  0xBF,  0x7D,  0xBD,  0xBC,  0x7C,  0xB4,  0x74,  0x75,  0xB5,
0x77,  0xB7,  0xB6,  0x76,  0x72,  0xB2,  0xB3,  0x73,  0xB1,  0x71,  0x70,  0xB0,  0x50,  0x90,  0x91,
0x51,  0x93,  0x53,  0x52,  0x92,  0x96,  0x56,  0x57,  0x97,  0x55,  0x95,  0x94,  0x54,  0x9C,  0x5C,
0x5D,  0x9D,  0x5F,  0x9F,  0x9E,  0x5E,  0x5A,  0x9A,  0x9B,  0x5B,  0x99,  0x59,  0x58,  0x98,  0x88,
0x48,  0x49,  0x89,  0x4B,  0x8B,  0x8A,  0x4A,  0x4E,  0x8E,  0x8F,  0x4F,  0x8D,  0x4D,  0x4C,  0x8C,
0x44,  0x84,  0x85,  0x45,  0x87,  0x47,  0x46,  0x86,  0x82,  0x42,  0x43,  0x83,  0x41,  0x81,  0x80,
0x40
}

unsigned short CRC16 (ptMsg, usDataLen)
unsigned char *ptMsg;                          / * message to calculate CRC upon * /
unsigned short usDataLen;                      / * number of bytes in message * /
{
        unsigned char uchCRCHi = 0xFF;         / * CRC high byte * /
        unsigned char uchCRCLo =0xFF;          / * CRC low byte * /
        unsigned uIndex;

        while (usDataLen--)                    / * pass through message buffer * /
        {
          uIndex = uchCRCHi ^ *ptMsg++;        /* calculate the CRC * /
          uchCRCHi = uchCRCLo ^ auchCRCHi [ uIndex ];
          uchCRCLo = auchCRCLo [ uIndex ]
        }
        return (uchCRCHi « 8 | uchCRCLo);
}
```

CRC CALCULATION (CHECKSUM for RTU)

Example of the CRC calculation:

FUNZIONE 03H: READ INPUT REGISTER

La funzione 03 permette di leggere una o più grandezze consecutive in memoria il cui indirizzo è indicato nelle tabelle nelle ultime pagine di questo manuale. Se l'indirizzo richiesto non è in tabella o il numero di registri richiesti è maggiore del consentito, l'AC-65 ritorna un messaggio di errore (vedi tabella errori).

Richiesta Master:

Indirizzo slave	01h
Funzione	03h
MSB indirizzo registro	01h
LSB indirizzo registro	00h
MSB numero registri	00h
LSB numero registri	10h
MSB CRC	45h
LSB CRC	FAh

Nell'esempio vengono richiesti, allo slave numero 1, 16 registri consecutivi a partire dall'indirizzo 0100h. Quindi vengono letti I registri dal 0100h al 010Fh. Il commando termina sempre con il valore checksum CRC.

FUNCTION 03H: READ INPUT REGISTER

The Modbus function 03 allows to read one or more consecutive registers. The address of each measures is given in the tables on the final page of this manual. If the measure address is not included in the table or the number of requested registers exceeds the max number, the AC-65 will return an error code (see error table).

Master query:

Slave address	01h
Function	03h
MSB register address	01h
LSB register address	00h
MSB register number	00h
LSB register number	10h
MSB CRC	45h
LSB CRC	FAh

In the above example, slave 1 is requested for 16 consevutive registers beginning with address 0100h. Thus, registers from 0100h to 010Fh will be returned. As usual, the message ends with CRC checksum.

Risposta Slave:

Indirizzo slave	01h
Funzione	03h
Numero di byte	20h
MSB dato 0100h	00h
LSB dato 0100h	01h
----------	----
MSB dato 010Fh	00h
LSB dato 010Fh	03h
MSB CRC	80h
LSB CRC	D7h

La.risposta è composta dall'indirizzo dello slave, dalla funzione richiesta dal Master e dai dati dei registri richiesti e termina sempre con il valore di checksum CRC.

Slave response:

Slave address	01h
Function	03h
Byte number	20h
MSB data register 0100h	00h
LSB data register 0100h	01h
----------	----
MSB data register 010Fh	00h
LSB data register 010Fh	03h
MSB CRC	80h
LSB CRC	D7h

The response is composed of the slave address, the function code requested by the master and the contents of the requested registers and ends with the CRC.

FUNZIONE 10H: PRESET MULTIPLE REGISTER

Questa funzione permette di modificare più parametri consecutivamente o parametri composti da più di 2 byte.

Richiesta Master:

Indirizzo slave	01h
Funzione	10h
MSB indirizzo registro	01h
LSB indirizzo registro	0Fh
MSB numero registri	00h
LSB numero registri	01h
Numero byte	02h
MSB Dato 010Fh	00h
LSB Dato 010Fh	03h
MSB CRC	F6h
LSB CRC	6Eh

Risposta Slave:

Indirizzo slave	01h
Funzione	10h
MSB indirizzo registro	01h
LSB indirizzo registro	0Fh
MSB numero byte	00h
LSB numero byte	01h
MSB CRC	30h
LSB CRC	36h

FUNCTION 10H: PRESET MULTIPLE REGISTER

This function allows to modify multiple parameters with a single message, or to preset a value longer that one register.

Master query:

Slave address	01h
Function	10h
MSB register address	01h
LSB register address	0Fh
MSB register number	00h
LSB register number	01h
Byte number	02h
MSB data	00h
LSB data	03h
MSB CRC	F6h
LSB CRC	6Eh

Slave response:

Slave address	01h
Function	10h
MSB register address	01h
LSB register address	0Fh
MSB byte number	00h
LSB byte number	01h
MSB CRC	30h
LSB CRC	36h

FUNZIONE 11H: REPORT SLAVE ID

Questa funzione permette di identificare il tipo di multimetro.

Richiesta Master:

Indirizzo slave	01h
Funzione	11h
MSB CRC	C0h
LSB CRC	2Ch

Risposta Slave:

Indirizzo slave	01h
Funzione	11h
Numero di byte	02h
Dato 1 (Tipo) ❶	67h
Dato 2 (Revisione software)	FFh
MSB CRC	D7h
LSB CRC	4Ch

❶ 67h = Compalarm A – AC65

FUNCTION 11H: REPORT SLAVE ID

This function allows to identify the multimeter type.

Master query:

Slave address	01h
Function	11h
MSB CRC	C0h
LSB CRC	2Ch

Slave response:

Slave address	01h
Function	11h
Byte number	02h
Data 1 (Type) ❶	67h
Data 2 (Software revision)	FFh
MSB CRC	D7h
LSB CRC	4Ch

❶ 67h = Compalarm A – AC65

ERRORI

Nel caso lo slave riceva un messaggio errato, segnala la condizione al master con un messaggio composto dalla funzione richiesta in OR con 80h, seguita da un codice di errore. Nella seguente tabella vengono riportati i codici di errore inviati dallo slave al master.

CODE	ERRORE
01	Funzione non valida
02	Indirizzo registro illegale
03	Valore del parametro fuori range

ERRORS

In case the slave receives an incorrect message, it answers with a message composed by the queried function OR with 80h, followed by an error code byte. In the following table are reported the error codes sent by the slave to the master.

CODE	ERROR
01	Invalid function
02	Invalid address
03	Parameter out of range

MISURE FORNITE DAL PROTOCOLLO DI COM.
(Utilizzabili con funzione 03H)

MEASURES SUPPLIED BY COMM. PROTOCOL
(To be used with function 03H)

INDIRIZZO/ADDRESS	WORDS	MISURA	MEASURE	FORMATO/FORMAT
0100H	1	Versione hardware	Hardware version	Unsigned int
0101H	1	Versione firmware	Firmware version	Unsigned int
0102H	1	Numero di ingressi	Number of inputs	Unsigned int
0103H	1	Stato casella d'allarme n°1 ❶	Alarm window n°1 status ❶	Unsigned int
0104H	1	Stato casella d'allarme n°2 ❶	Alarm window n°2 status ❶	Unsigned int
0105H	1	Stato casella d'allarme n°3 ❶	Alarm window n°3 status ❶	Unsigned int
0106H	1	Stato casella d'allarme n°4 ❶	Alarm window n°4 status ❶	Unsigned int
0107H	1	Stato casella d'allarme n°5 ❶	Alarm window n°5 status ❶	Unsigned int
0108H	1	Stato casella d'allarme n°6 ❶	Alarm window n°6 status ❶	Unsigned int
0109H	1	Stato ingressi ❷	Alarm inputs status ❷	Unsigned int
010AH	1	Stato relè allarme cumulativo ❸	Cumulative output relay status ❸	Unsigned int
010BH	1	Stato relè sirena ❹	Audible output relay status ❹	Unsigned int
010CH	1	Impostazione NO/NC ingressi ❺	NO/NC inputs setting ❺	Unsigned int
010DH	1	Abilitazione ingressi First-out ❻	First-out enable inputs setting ❻	Unsigned int
010EH	1	Sequenza d'allarme ❼	Alarm sequence ❼	Unsigned int

❶ Stato casella d'allarme
0 = spenta 2 = lampeggio lento 4 = lampeggio irregolare
1 = fissa 3 = lampeggio veloce

❶ Alarm window
0 = off 2 = slow flashing 4 = intermittent flashing
1 = on 3 = fast flashing

❷ Stato ingressi
Esempio: Il valore all'indirizzo 0109H è 0x05 (esadecimale), = 00000101 vuol dire che gli ingressi 1 e 3 sono impegnati. Bit=0: ingresso non impegnato. Bit=1 ingresso impegnato.

❷ Alarm inputs status
Example: The value at address 0109H is 0x05 (hexadecimal), = 00000101 means that the inputs 1 and 3 are active. Bit=0 indicates a not active input. Bit=1indicates an active input.

❸ Stato relè allarme cumulativo
0 = Nessun allarme 1 = Almeno un allarme presente

❸ Stato relè allarme cumulativo
0 = No alarm 1 = At least one alarm present

❹ Stato relè sirena
0 = off 1 = on

❹ Audible output relay status
0 = off 1 = on

❺ Impostazione NO/NC ingressi
0 = Normalmente Aperto (NO) 1 = Normalmente Chiuso (NC)

❺ NO/NC inputs setting
0 = Normally Open (NO) 1 = Normally Close (NC)

❻ Impostazione abilitazione ingressi First-out
0 = First-out non abilitato 1 = First-out abilitato
Esempio: Il valore all'indirizzo 010DH è 0x05 (esadecimale), = 100110 vuol dire che gli ingressi 2,3 e 6 hanno il Fisrt-out abilitato.

❻ First-out enable inputs setting
0 = First-out not enable 1 = First-out enable
Example: The value at address 010DH is 0x05 (hexadecimal), = 00100101 means that the inputs 2,3 and 6 have First-out enable

❼ Sequenza d'allarme
0 = F1M 1 = F3A 2 = F1A 3 = M 4 = R8 5 = M5 6 = A

❼ Alarm sequence
0 = F1M 1 = F3A 2 = F1A 3 = M 4 = R8 5 = M5 6 = A

PARAMETRI SETUP
(Utilizzabili con funzioni 03H e 10H)

SETUP PARAMETERS
(To be used with functions 03H and 10H)

INDIRIZZO/ADDRESS	WORDS	MENU	MENU	MIN	MAX	DEF
		Comunicazione	**Communication**			
010FH	1	Indirizzo seriale nodo	Serial node address	1	247	1
0110H	1	Velocità seriale ❶	Serial speed ❶	1	6	3

❶ Velocità seriale
1 = 4800bps 2 = 9600bps 3 = 19200bps 4 = 38400bps 5 = 57600bps 6 = 115200bps

❶ Serial speed

COMANDI
(Utilizzabili con funzioni 10H)

COMMANDS
(To be used with functions 10H)

INDIRIZZO/ADDRESS	WORDS	COMANDO	COMMAND	UNITA'/UNIT	FORMATO/FORMAT
0111H	1	Comando ACK ❶	ACK command ❶	-	Unsigned int
0112H	1	Comando RESET ❶	RESET command ❶	-	Unsigned int

❶ Impostare '1' per effettuare il comando, '0' nessuna azione

❶ Set '1' to trigger, '0' no action

Per ulteriori informazioni contattare:

For further details please contact:

Contrel elettronica s.r.l.
Via San Fereolo, 9
I-26900 Lodi
Tel: +39 0371 30207 / 30761 / 35386
Fax: +39 0371 32819
E-Mail: contrel@contrel.eu

www.contrel.it