31100257

I454IGB03 21

LOVATO ELECTRIC S.P.A.
24020 GORLE (BERGAMO) ITALIA
VIA DON E. MAZZA, 12
TEL. 035 4282111
TELEFAX (Nazionale): 035 4282200
TELEFAX (International): +39 035 4282400
Web www.LovatoElectric.com
E-mail info@LovatoElectric.com

(I)

SOFT STARTER

ADXL

PROTOCOLLO DI COMUNICAZIONE MODBUS®

ADDENDUM

(GB)

SOFT STARTER

ADXL

MODBUS COMUNICATION PROTOCOL®

ADDENDUM

CE

PROTOCOLLO MODBUS®

I soft starters della serie ADXL supportano i protocolli di comunicazione Modbus RTU®, Modbus ASCII® e Modbus TCP®.
Per consentire la comunicazione, i soft starters ADXL devono essere equipaggiati con l'interfaccia di comunicazione RS-485, codice EXC1042 (da acquistare separatamente).

Grazie a questa funzione è possibile leggere lo stato degli apparecchi e controllare gli stessi tramite software di supervisione Lovato Electric (Synergy e Xpress) o software standard forniti da terze parti (SCADA) oppure tramite apparecchiature dotate di interfaccia Modbus® quali PLC e terminali intelligenti.

IMPOSTAZIONE DEI PARAMETRI

Per configurare il protocollo Modbus®, accedere al menu 01-SETUP e selezionare il menu P08-COMUNICAZIONE: è possibile configurare una sola porta di comunicazione.

Comunicazione seriale

PAR	Funzione	Default	Range
08.01	Indirizzo seriale nodo	1	1-255
08.02	Velocità seriale	9600	1200 2400 4800 9600 19200 38400 57600 115200
08.03	Formato dati	8 bit – n	8 bit, no parità 8 bit, dispari 8 bit, pari 7 bit, dispari 7 bit, pari
08.04	Bit di stop	1	1-2
08.05	Protocollo	Modbus RTU	Modbus RTU Modbus ASCII Modbus TCP

MODBUS® PROTOCOL

The ADXL series of soft starters support the communication protocols Modbus RTU®, Modbus ASCII® and Modbus TCP®.
To allow the communication the soft starters ADXL must be equipped with the RS-485 communication board code EXC1042 (to be purchased separately).

Using this function it is possible to read the device status and to control the units through Lovato Electric software (Synergy and Xpress) or third-party supervision software (SCADA) or through other intelligent devices supporting Modbus®, like PLCs.

PARAMETERS SETTING

To configure the Modbus® protocol, enter in the menu 01-SETUP and then go to the menu P08-COMMMUNICATION: it is possible to configure only one communication port.

Serial comunication

PAR	Function	Default	Range
08.01	Serial node address	1	1-255
08.02	Baudrate	9600	1200 2400 4800 9600 19200 38400 57600 115200
08.03	Data format	8 bit – n	8 bit –no par. 8 bit, odd 8 bit, even 7 bit, odd 7 bit, even
08.04	Stop bits	1	1-2
08.05	Protocol	Modbus RTU	Modbus RTU Modbus ASCII Modbus TCP

PROTOCOLLO MODBUS® RTU

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

T1 T2 T3	Indirizzo (8 bit)	Funzione (8 bit)	Dati (N x 8 bit)	CRC (16 bit)	T1 T2 T3

- 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. Questo consente, in caso di disturbo sulla linea di trasmissione, di ignorare il messaggio inviato per evitare problemi sia dal lato master sia slave.
- La sequenza T1 T2 T3 corrisponde al tempo durante il quale non devono essere scambiati dati sul bus di comunicazione, per consentire agli strumenti collegati di riconoscere la fine di un messaggio e l'inizio del successivo. Questo tempo deve essere pari a 3.5 caratteri.

Il dispositivo misura il tempo trascorso tra la ricezione di un carattere e il successivo e se questo tempo supera quello necessario per trasmettere 3.5 caratteri, riferiti al baud rate impostato, il prossimo carattere viene considerato l'inizio di un nuovo messaggio.

FUNZIONI MODBUS®

Le funzioni disponibili sono:

03 = Read input register	Consente la lettura delle misure disponibili sul dispositivo
04 = Read input register	Consente la lettura delle misure disponibili sul dispositivo
06 = Preset single register	Permette la scrittura dei parametri
07 = Read exception	Permette di leggere lo stato dell'apparecchio
17 = Report slave ID	Permette di leggere informazioni relative all'apparecchio

Per esempio, se si vuole leggere dal soft starter ADXL con indirizzo 01 il valore della temperatura dei tiristori, che si trova alla locazione 0FB2 Hex, il messaggio da spedire è il seguente:

01	04	0F	B1	00	02	22	F8

Dove:
01= Indirizzo slave.
04 = Funzione di lettura locazione.
0FB1 = Indirizzo della locazione diminuito di un'unità, contenete il valore della temperatura dei tiristori.
0002 = Numero di registri da leggere a partire dall'indirizzo 0FB1.
22 F8 = Checksum CRC.

La risposta del dispositivo è la seguente:

01	04	04	00	00	01	10	3B	C3

Dove:
01= indirizzo del dispositivo (Slave 01).
04 = funzione richiesta dal Master.
04 = numero di byte inviati dal dispositivo.
00 00 01 10 = valore esadecimale della temperatura = 272 → 27.2 °C.
3B C3 = checksum CRC.

FUNZIONE 04: READ INPUT REGISTER

La funzione 04 permette di leggere una o più grandezze consecutive in memoria. L'indirizzo di ciascuna grandezza è indicato nella Tabella 2. Come da standard Modbus®, l'indirizzo specificato nel messaggio va diminuito di 1 rispetto a quello effettivo riportato nella tabella. Se l'indirizzo richiesto non è compreso nella tabella o il numero di registri richiesti è maggiore del numero consentito il dispositivo ritorna un messaggio di errore (vedi tabella errori).

MODBUS® RTU PROTOCOL

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

T1 T2 T3	Address (8 bit)	Function (8 bit)	Data (N x 8 bit)	CRC (16 bit)	T1 T2 T3

- 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 slave 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. If a message has been corrupted by electrical noise or interference, the CRC field allows the devices to recognize the error and thereby to ignore the message.
- The T1 T2 T3 sequence corresponds to a time in which data must not be exchanged on the communication bus to allow the connected devices to recognize the end of one message and the beginning of another. This time must be at least 3.5 times the time required to send one character.

The device measures the time that elapses from the reception of one character and the following. If this time exceeds the time necessary to send 3.5 characters at the selected baudrate, then the next character will be considered as the first of a new message.

MODBUS® FUNCTIONS

The available functions are:

03 = Read input register	Allows to read the device measures.
04 = Read input register	Allows to read the device measures.
06 = Preset single register	Allows writing parameters
07 = Read exception	Allows to read the device status
17 = Report slave ID	Allows to read information about the device.

For instance, to read the value of the temperature of the thyristors, which resides at location 0FB2 Hex, from the ADXL with serial address 01, the message to send is the following:

01	04	0F	B1	00	02	22	F8

Where:
01= Slave address.
04 = Modbus® function 'Read input register'.
0FB1 = Address of the required register (which contains the value of the thyristors temperature) decreased by one.
00 02 = Number of registers to be read beginning from address 2D.
22 F8 = CRC Checksum.

The device answer is the following:

01	04	04	00	00	01	10	3B	C3

Where:
01 = device address (Slave 01).
04 = Function requested by the master.
04 = Number of bytes sent by the device.
00 00 01 10 = Hex value of the temperature = 272 → 27.2 °C.
3B C3 = CRC checksum.

FUNCTION 04: READ INPUT REGISTER

The Modbus® function 04 allows to read one or more consecutive registers from the slave memory. The address of each measure is given in the table 2. As for Modbus® standard, the address in the query message must be decreased by one from the effective address reported in the table. If the measure address is not included in the table or the number of requested registers exceeds the acceptable max number, the device will return an error code (see error table).

Richiesta Master:

Indirizzo slave	01h
Funzione	04h
MSB Indirizzo registro	00h
LSB Indirizzo registro	0Fh
MSB Numero registri	00h
LSB Numero registri	08h
LSB CRC	C1h
MSB CRC	CFh

Nell'esempio vengono richiesti, allo slave numero 1, 8 registri consecutivi a partire dall'indirizzo 10h. Quindi vengono letti i registri dall' 10h al 17h. Il comando termina sempre con il valore di checksum CRC.

Risposta Slave:

Indirizzo slave	01h
Funzione	04h
Numero di byte	10h
MSB Dato 10h	00h
LSB Dato 10h	00h
----------	----
MSB Dato 17h	00h
LSB Dato 17h	00h
LSB CRC	55h
MSB CRC	2Ch

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

FUNZIONE 06: PRESET SINGLE REGISTER
Questa funzione permette di scrivere nei registri. E' possibile ad esempio impostare i parametri del setup. Qualora il valore impostato non rientri nel valore minimo e massimo della tabella il dispositivo risponderà con un messaggio di errore. Se viene richiesto un parametro ad un indirizzo inesistente verrà risposto con un messaggio di errore.

Richiesta Master:

Indirizzo slave	01h
Funzione	06h
MSB Indirizzo registro	4Fh
LSB Indirizzo registro	FFh
MSB Dato	00h
LSB Dato	01h
LSB CRC	6Eh
MSB CRC	EEh

Risposta Slave: la risposta è un eco della domanda, cioè viene inviato al master l'indirizzo del dato da modificare e il nuovo valore del parametro.

FUNZIONE 07: READ EXCEPTION STATUS
Tale funzione permette di leggere lo stato in cui si trova il soft starter.

Richiesta Master:

Indirizzo slave	01h
Funzione	07h
LSB CRC	41h
MSB CRC	E2h

La tabella seguente riporta il significato del byte inviato dal ADXL come risposta:

BIT	SIGNIFICATO
0	Ingresso start chiuso
1	Ingresso stop chiuso
2	Condizione di blocco attiva
3	Condizioni per avvio presenti
4	Condizioni per avvio presenti + start attivo
5	Allarme attivo
6	Motore in marcia
7	Uscita allarme globale attiva
8	Bypass attivo

Master query:

Slave address	01h
Function	04h
MSB address	00h
LSB address	0Fh
MSB register number	00h
LSB register number	08h
LSB CRC	C1h
MSB CRC	CFh

In the above example, slave 01 is requested for 8 consecutive registers beginning with address 10h. Thus, registers from 10h to 17h will be returned. As usual, the message ends with the CRC checksum.

Slave response:

Slave address	01h
Function	04h
Byte number	10h
MSB register 10h	00h
LSB register 10h	00h
----------	----
MSB register 17h	00h
LSB register 17h	00h
LSB CRC	55h
MSB CRC	2Ch

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

FUNCTION 06: PRESET SINGLE REGISTER
This function allows to write in the registers. For instance, it is possible to change setup parameters. If the value is not in the correct range, the device will answer with an error message. In the same way, if the parameter address is not recognised, the device will send an error response.

Master message:

Indirizzo slave	01h
Funzione	06h
MSB Indirizzo registro	4Fh
LSB Indirizzo registro	FFh
MSB Dato	00h
LSB Dato	01h
LSB CRC	6Eh
MSB CRC	EEh

Slave response: the slave response is an echo to the query, that is the slave sends back to the master the address and the new value of the variable.

FUNCTION 07: READ EXCEPTION STATUS
This function allows to read the status of the power factor controller.

Master query:

Slave address	01h
Function	07h
LSB CRC	41h
MSB CRC	E2h

The following table gives the meaning of the status byte sent by the ADXL as answer:

BIT	MEANING
0	Start input closed
1	Stop input closed
2	Block condition activated
3	Start conditions present
4	Start conditions present + start active
5	Alarm active
6	Motor running
7	Global alarm output active
8	Bypass active

FUNZIONE 17: REPORT SLAVE ID
Questa funzione permette di identificare il tipo di dispositivo.

Richiesta Master.

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

Risposta Slave:

Indirizzo slave	01h
Funzione	11h
Contatore bytes	08h
Dato 01 (tipo) ❶	01h
Dato 02 (revisione software)	02h
Dato 03 (revisione hardware)	00h
Dato 04 (revisione parametri)	00h
Dato 05 (tipoolgia di prodotto) ❷	05h
Dato 06 (riservato)	00h
Dato 07 (riservato)	00h
Dato 08 (riservato)	00h
LSB CRC	86h
MSB CRC	88h

❶
1 - 01h = ADXL0030600
2 - 02h = ADXL0045600
3 - 03h = ADXL0060600
4 - 04h = ADXL0075600
5 - 05h = ADXL0085600
6 - 06h = ADXL0115600
7 - 07h = ADXL0135600
8 - 08h = ADXL0162600
9 - 09h = ADXL0195600
10 - 0Ah = ADXL0250600
11 – 0Bh = ADXL0320600

❷
5 – 05h= Serie AXDL

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

TABELLA 1: CODICI ERRORE

COD	ERRORE
01	Funzione non valida
02	Indirizzo registro illegale
03	Valore del parametro fuori range
04	Impossibile effettuare operazione
06	Slave occupato, funzione momentaneamente non disponibile

FUNZIONE 17: REPORT SLAVE ID
This function allows to identify the device type.

Master query.

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

Slave response:

Slave address	01h
Function	11h
Byte count	08h
Data 01 (type)❶	01h
Data 02 (software revision)	02h
Data 03 (hardware revision)	00h
Data 04 (parameter revision)	00h
Data 05 (type of device) ❷	05h
Data 06 (reserved)	00h
Data 07 (reserved)	00h
Data 08 (reserved)	00h
LSB CRC	86h
MSB CRC	88h

❶
1 - 01h = ADXL0030600
2 - 02h = ADXL0045600
3 - 03h = ADXL0060600
4 - 04h = ADXL0075600
5 - 05h = ADXL0085600
6 - 06h = ADXL0115600
7 - 07h = ADXL0135600
8 - 08h = ADXL0162600
9 - 09h = ADXL0195600
10 - 0Ah = ADXL0250600
11 – 0Bh = ADXL0320600

❷
5 – 05h= AXDL series

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

TABLE 1: ERROR CODES

CODE	ERROR
01	Invalid function
02	Invalid address
03	Parameter out of range
04	Function execution impossible
06	Slave busy, function momentarily not available

PROTOCOLLO MODBUS® ASCII

Il protocollo Modbus® ASCII viene utilizzato normalmente nelle applicazioni che richiedono di comunicare via modem. Le funzioni e gli indirizzi disponibili sono gli stessi della versione RTU, ma i caratteri trasmessi sono in ASCII e la terminazione del messaggio non è effettuata a tempo ma con dei caratteri di ritorno a capo.
Se si seleziona il parametro 08.05 come protocollo Modbus® ASCII, la struttura del messaggio di comunicazione sulla relativa porta di comunicazione è così costituita:

:	Indirizzo 2 chars	Funzione 2 chars	Dati (N chars)	LRC 2 chars	CR LF

- 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 LRC consente sia al master che allo slave di verificare se ci sono errori di trasmissione. Questo consente, in caso di disturbo sulla linea di trasmissione, di ignorare il messaggio inviato per evitare problemi sia dal lato master che slave.
- Il messaggio termina sempre con i caratteri di controllo CRLF (0D 0A).

Esempio:

Per esempio, se si vuole leggere dal dispositivo con indirizzo 01 il valore della temperatura che si trova alla locazione 0FB2h, il messaggio da spedire è il seguente:

:	01	04	0F	B1	00	02	39	CRLF

Dove:
: = ASCII 3Ah = Delimitatore inizio messaggio
01 = indirizzo slave.
04 = funzione di lettura locazione.
0F B1 = indirizzo della locazione (temperatura) diminuito di un'unità.
00 02 = numero di registri da leggere a partire dall'indirizzo 04.
39 = checksum LRC.
CRLF = ASCII 0Dh 0Ah = delimitatore fine messaggio

La risposta del dispositivo è la seguente:

:	01	04	04	00	00	00	FF	F8	CRLF

Dove:
: = ASCII 3Ah = Delimitatore inizio messaggio.
01 = Indirizzo del dispositivo (Slave 01).
04 = Funzione richiesta dal Master.
04 = Numero di byte inviati dallo slave.
00 00 00 FF = Valore esadecimale della temperatura misurata (=25.5°C).
F8 = Checksum LRC.
CRLF = ASCII 0Dh 0Ah = Delimitatore fine messaggio

MODBUS® ASCII PROTOCOL

The Modbus® ASCII protocol is normally used in application that require to communicate through a couple of modems. The functions and addresses available are the same as for the RTU version, but the transmitted characters are in ASCII and the message end is delimited by Carriage return / Line Feed instead of a transmission pause. If one selects the parameter 08.05 as Modbus® ASCII protocol, the communication message on the correspondent communication port has the following structure:

:	Address (2 chars)	Function (2 chars)	Dates (N chars)	LRC (2 chars)	CR LF

- 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 slave or data received from the slave in response to a query.
- The LRC field allows the master and slave devices to check the message integrity. If a message has been corrupted by electrical noise or interference, the LRC field allows the devices to recognize the error and thereby ignore the message.
- The message terminates always with CRLF control character (0D 0A).

Example:

For instance, to read the value of the temperature, which resides at location 0FB2h from the slave with serial address 01, the message to send is the following:

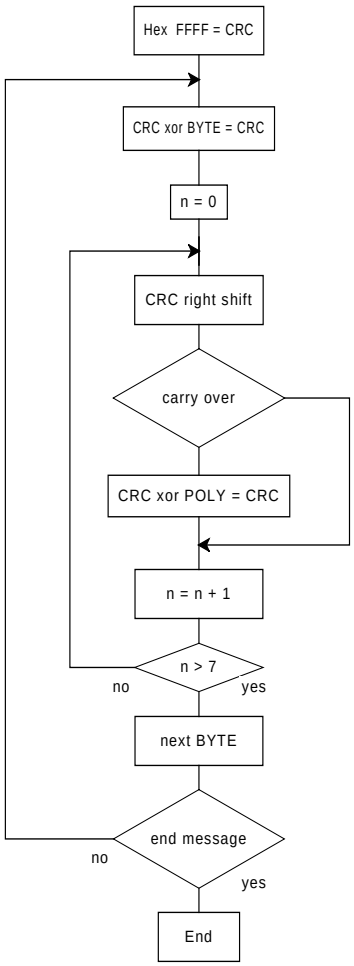
:	01	04	0F	B1	00	02	39	CRLF

Whereas:
: = ASCII 3Ah message start delimiter
01 = slave address.
04 = Modbus® function 'Read input register'
0F B1 = Address of the required register (temperature) decreased by one.
00 02 = Number of registers to be read beginning from address 04.
39 = LRC Checksum.
CRLF = ASCII 0Dh 0Ah = Message end delimiter

The answer of the device is the following:

:	01	04	04	00	00	00	FF	F8	CRLF

Where:
: = ASCII 3Ah message start delimiter.
01 = ADXL address (Slave 01).
04 = Function requested by the master.
04 = Number of bytes sent by the slave.
00 00 00 FF = Hex value of the measured temperature (=25.5°C).
F8 = LRC checksum
CRLF = ASCII 0Dh 0Ah = Message end delimiter.



CALCOLO DEL CRC (CHECKSUM per RTU)
Esempio di calcolo:
Frame = 0207h

Inizializzazione CRC	1111	1111	1111	1111	
Carica primo byte			0000	0010	
Esegue xor con il primo Byte della frame	1111	1111	1111	1101	
Esegue primo shift a dx	0111	1111	1111	1110	1
Carry=1, carica polinomio	1010	0000	0000	0001	
Esegue xor con il polinomio	1101	1111	1111	1111	
Esegue secondo shift dx	0110	1111	1111	1111	1
Carry=1,carica polinomio	1010	0000	0000	0001	
Esegue xor con il polinomio	1100	1111	1111	1110	
Esegue terzo shift	0110	0111	1111	1111	0
Esegue quarto shift	0011	0011	1111	1111	1
Carry=1, carica polinomio	1010	0000	0000	0001	
Esegue xor con il Polinomio	1001	0011	1111	1110	
Esegue quinto shift dx	0100	1001	1111	1111	0
Esegue sesto shift dx	0010	0100	1111	1111	1
Carry=1, carica polinomio	1010	0000	0000	0001	
Esegue xor con polinomio	1000	0100	1111	1110	
Esegue settimo shift dx	0100	0010	0111	1111	0
Esegue ottavo shift dx	0010	0001	0011	1111	1
Carry=1, carica polinomio	1010	0000	0000	0001	
Carica secondo byte della frame			0000	0111	
Esegue xor con il Secondo byte della frame	1000	0001	0011	1001	
Esegue primo shift dx	0100	0000	1001	1100	1
Carry=1, carica polinomio	1010	0000	0000	0001	
Esegue xor con il polinomio	1110	0000	1001	1101	
Esegue secondo shift dx	0111	0000	0100	1110	1
Carry=1, carica polinomio	1010	0000	0000	0001	
Esegue xor con il polinomio	1101	0000	0100	1111	
Esegue terzo shift dx	0110	1000	0010	0111	1
Carry=1, carica polinomio	1010	0000	0000	0001	
Esegue xor con il polinomio	1100	1000	0010	0110	
Esegue quarto shift dx	0110	0100	0001	0011	0
Esegue quinto shift dx	0010	0100	0000	1001	1
Carry=1, carica polinomio	1010	0000	0000	0001	
Esegue xor con il polinomio	1001	0010	0000	1000	
Esegue sesto shift dx	0100	1001	0000	0100	0
Esegue settimo shift dx	0010	0100	1000	0010	0
Esegue ottavo shift dx	0001	0010	0100	0001	0
Risultato CRC	0001	0010			
0100	0001				
	12h	41h			

Nota: Il byte 41h viene spedito per primo (anche se è il LSB), poi viene trasmesso 12h.

CALCOLO LRC (CHECKSUM per ASCII)
Esempio di calcolo:

Indirizzo	01	00000001
Funzione	04	00000100
Start address hi.	00	00000000
Start address lo.	00	00000000
Numero registri	08	00001000
Somma		00001101
Complemento a 1		11110010
+ 1		00000001
Complemento a 2		11110101
Risultato LRC		
		F5

CRC CALCULATION (CHECKSUM for RTU)
Example of CRC calculation:
Frame = 0207h

CRC initialization	1111	1111	1111	1111	
Load the first byte			0000	0010	
Execute xor with the first Byte of the frame	1111	1111	1111	1101	
Execute 1st right shift	0111	1111	1111	1110	1
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1101	1111	1111	1111	
Execute 2nd right shift	0110	1111	1111	1111	1
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1100	1111	1111	1110	
Execute 3rd right shift	0110	0111	1111	1111	0
Execute 4th right shift	0011	0011	1111	1111	1
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1001	0011	1111	1110	
Execute 5th right shift	0100	1001	1111	1111	0
Execute 6th right shift	0010	0100	1111	1111	1
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1000	0100	1111	1110	
Execute 7th right shift	0100	0010	0111	1111	0
Execute 8th right shift	0010	0001	0011	1111	1
Carry=1,load polynomial	1010	0000	0000	0001	
Load the second byte of the frame			0000	0111	
Execute xor with the Second byte of the frame	1000	0001	0011	1001	
Execute 1st right shift	0100	0000	1001	1100	1
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1110	0000	1001	1101	
Execute 2nd right shift	0111	0000	0100	1110	1
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1101	0000	0100	1111	
Execute 3rd right shift	0110	1000	0010	0111	1
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1100	1000	0010	0110	
Execute 4th right shift	0110	0100	0001	0011	0
Execute 5th right shift	0010	0100	0000	1001	1
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1001	0010	0000	1000	
Execute 6th right shift	0100	1001	0000	0100	0
Execute 7th right shift	0010	0100	1000	0010	0
Execute 8th right shift	0001	0010	0100	0001	0
CRC Result	0001	0010		0100	
	0001				
	12h	41h			

Note: The byte 41h is sent first(even if it is the LSB), then12h is sent.

LRC CALCULATION (CHECKSUM for ASCII)
Example of LRC calculation:

Address	01	00000001
Function	04	00000100
Start address hi.	00	00000000
Start address lo.	00	00000000
Number of registers	08	00001000
Sum		00001101
1. complement		11110010
+ 1		00000001
2. complement		11110101
LRC result		
		F5

TABELLA 2:
MISURE FORNITE DAL PROTOCOLLO DI COMUNICAZIONE
(Utilizzabili con funzioni 03 e 04)

TABLE 2:
MEASURES SUPPLIED BY SERIAL COMMUNICATION PROTOCOL
(To be used with functions 03 and 04)

INDIRIZZO ADDRESS	WORDS	MISURA	MEASURE	UNITA' UNIT	FORMATO FORMAT
06h	2	Tensione L3-L1	L3-L1 Voltage	V / 100	Unsigned long
08h	2	Corrente L1	L1 current	A / 10000	Unsigned long
0Ah	2	Corrente L2	L2 current	A / 10000	Unsigned long
0Ch	2	Corrente L3	L3 current	A / 10000	Unsigned long
14h	2	Potenza attiva L1	L1 active power	kW / 100000	Signed long
16h	2	Potenza attiva L2	L2 active power	kW / 100000	Signed long
18h	2	Potenza attiva L3	L3 active power	kW / 100000	Signed long
26h	2	Fattore di potenza fase L1	L1 power factor	/ 10000	Signed long
28h	2	Fattore di potenza fase L2	L2 power factor	/ 10000	Signed long
2Ah	2	Fattore di potenza fase L3	L3 power factor	/ 10000	Signed long
32h	2	Frequenza	Frequency	Hz / 1000	Unsigned long
3Ah	2	Potenza attiva totale	Total active power	kW / 10000	Signed long
40h	2	Fattore di potenza totale	Total power factor	/ 10000	Signed long
76h	2	Corrente massima	Max current	A / 10000	Unsigned long
78h	2	Coppia	Torque	%/ 10	Unsigned long
7Ah	2	Corrente istantanea massima %	Maximum instantaneous current %	% / 10	Unsigned long
F80h	2	Contaore totale	Motor hour meter	h / 3600	Unsigned long
F82h	2	Numero di avviamenti	Start counter		Unsigned long
F84h	2	Stato ❶	Status		Unsigned integer
FB0h	2	Stato termico motore	Motor thermal status	%	Unsigned long
FB2h	2	Temperatura tiristori	Thyristors temperature	°C / 10	Unsigned long
1B20h	4	Energia attiva totale	Total active energy	kWh / 100	Unsigned long
2100h – bit 0	1	Ingresso 1	Input 1	bool	Unsigned integer
2100h– bit 1	1	Ingresso 2	Input 2	bool	Unsigned integer
2100h– bit 2	1	Ingresso 3	Input 3	bool	Unsigned integer
2141h – bit 0	1	Uscita 1	Output 1	bool	Unsigned integer
2141h – bit 1	1	Uscita 2	Output 2	bool	Unsigned integer
2141h – bit 2	1	Uscita 3	Output 3	bool	Unsigned integer
2180h - bit 0	1	Remoto 1	Remote 1	bool	Unsigned integer
2180h – bit 1	1	Remoto 2	Remote 2	bool	Unsigned integer
2180h – bit 2	1	Remoto 3	Remote 3	bool	Unsigned integer
2180h – bit3	1	Remoto 4	Remote 4	bool	Unsigned integer
21C0h – bit 0	1	Limite 1	Limit 1	bool	Unsigned integer
21C0h – bit 1	1	Limite 2	Limit 2	bool	Unsigned integer
21C0h – bit 2	1	Limite 3	Limit 3	bool	Unsigned integer
21C0h – bit 3	1	Limite 4	Limit 4	bool	Unsigned integer

❶ Significato del valore della risposta del registro di stato – Meaning of the answer of the status register:

VALUE	SIGNIFICATO	MEANING
0	Mancanza linea	Absence of power
1	Preriscaldo	Preheating
2	Avviatore pronto	Starter ready
3	Ritardo start	Start delay
4	Kick avviamento	Kick start
5	Rampa accelerazione	Acceleration ramp
6	Limite corrente	Current limit
7	Limite coppia	Torque limit
8	Marcia	Run
9	Bypass chiuso	Bypass closed
10	Rampa decelerazione	Deceleration ramp
11	Protezioni inibite	Protections inhibited
12	Ruota libera	Freewheel
13	Allarme	Alarm
14	Selezione motore	Motor selection

TABELLA 3:
COMANDI
(Utilizzabili con funzione 06)

TABLE 3:
COMMANDS
(To be used with function 06)

INDIRIZZO ADDRESS	WORDS	COMANDO	COMMAND	VALORE VALUE	FORMATO FORMAT
2FF0h	1	C01 Azzera intervallo di manutenzione.	*C01 Reset maintenance service interval.*	0	Unsigned int
2FF0h	1	C02 Reset stato termico.	*C02 Thermal status reset.*	1	Unsigned int
2FF0h	1	C03 Azzera contatore avviamenti.	*C03 Reset the number of startings counter.*	2	Unsigned int
2FF0h	1	C04 Azzera contaore motore.	*C04 Reset the motor's hour meter.*	3	Unsigned int
2FF0h	1	C05 Azzera contatori energia.	*C05 Reset the energy counters.*	4	Unsigned int
2FF0h	1	C06 Azzera variabili LIM con memoria.	*C06 Reset LIM variables with memory.*	5	Unsigned int
2FF0h	1	C11 Ripete procedura guidata AUTOSET.	*C11 Repeat the AUTOSET wizard.*	❶ 10	Unsigned int
2FF0h	1	C12 Mette setup a default di fabbrica.	*C12 Restore the factory default settings.*	❶ 11	Unsigned int
2FF0h	1	C13 Salva una copia dei parametri di setup.	*C13 Save a copy of the setup parameters.*	❶ 12	Unsigned int
2FF0h	1	C14 Ripristina copia dei parametri di setup.	*C14 Restore a copy of the setup parameters.*	❶ 13	Unsigned int
2FF0h	1	C15 Test con motore di bassa potenza	*C15 Test with low power motor.*	14	Unsigned int
2FF0h	1	C16 Azzera la memoria della lista eventi.	*C16 Reset the events list.*	15	Unsigned int
1002h	1	Comando di marcia motore	*Motor start command*	1	Unsigned int
1003h	1	Comando di arresto motore	*Motor stop command*	1	Unsigned int

❶ ATTENZIONE: dopo aver usato questo comando è preferibile inviare il comando di REBOOT (scrivere con funzione 06 il valore 5 all'indirizzo 2F03h).
ATTENTION: after using of this command it is recommended to send REBOOT command (write with function 06 the value 5 at the address 2F03h).

TABELLA 4:
EVENTI

TABLE 4:
EVENTS

INDIRIZZO *ADDRESS*	WORDS	MISURA	MEASURE	UNITA' UNIT	FORMATO FORMAT
5030h	1	PUNTATORE EVENTI indica l'ultimo evento registrato (LSB) / CONTATORE EVENTI indica il numero totale eventi (MSB)	EVENTS POINTER last event stored (LSB)/ EVENTS COUNTER total events stored (MSB)		Unsigned int
5032h	28	Descrizione evento nella lingua corrente	Event description using current language		Unsigned int
Procedura per la lettura degli eventi 1 - Eseguire la lettura di 1 registro con la funzione 04 all' indirizzo 5030h, il byte più significativo (MSB) indica quanti eventi sono memorizzati (valore compreso tra 0 e 60), il byte meno significativo (LSB) viene incrementato ogni volta che un evento viene salvato (valore compreso tra 0 e 60). Una volta memorizzati 60 eventi l'MSB resterà a 60 mentre l'LSB tornerà a zero e poi continuerà ad incrementare. 2- Impostare l'indice dell'evento che si vuole leggere (minore del numero massimo di eventi memorizzati),per fare questo bisogna eseguire la funzione 06 all'indirizzo 5030h, specificando quale evento leggere. 3- Eseguire una lettura di 28 registri (con un'unica funzione 04) all'indirizzo 5032h. Il valore tornato è una stringa di caratteri ASCII, che riportano la stessa descrizione dell'evento visibile sul display del ADXL. L'indice dell'evento che si vuole leggere viene incrementato in automatico dopo la lettura del registro 5032h,al fine di velocizzare il download degli eventi 4-Se si vuole leggere l'evento successivo eseguire il punto 3,se si vuole leggere un qualsiasi altro evento eseguire il passo 2.			Procedure for events reading 1 - Perform the read of 1 register by using the function 04 at address 5030h, the most significant byte (MSB) indicates how many events are stored (value between 0 to 60), the least significant byte (LSB) is incremented each time an event is saved (value between 0 to 60). Once stored the 60 events the MSB will remain at 60 while the LSB will back to zero and after will continue to increase. 2- Set the index of the event that you want to read (less than the maximum number of events stored), to do this performe the function 06 at 5030h, specifying which event read. 3- Perform a read of 28 registers (with a single function 04) at address 5032h. The value returned is a string of ASCII characters, showing the same event description visible on the display of the ADXL. The index of the event to be read is incremented automatically after a reading of the register 5032h, in order to speed up the download of events 4- If you want to read the next event performing step 3, if you want to read any other event do step 2.		

IMPOSTAZIONE PARAMETRI - *PARAMETER SETTING*
I parametri del ADXL vengono letti/modificati applicando la seguente regola. *ADXL parameters are read/modified according to the following rules.*

INDIRIZZO *ADDRESS*	WORDS	SIGNIFICATO *MEANING*	FUNZIONE *FUNCTION*	ESEMPIO *EXAMPLE*
5000h	1	Selezione numero menu Menu number selection	4 read – 6 write	Per selezionare il menu 1 scrivere il valore 1. Write value 1 to select the menu number 1.
5001h	1	Selezione numero sotto-menu (se presente) Sub-menu number selection (if present)	4 read – 6 write	Per selezionare il sotto-menu 1 scrivere il valore 1. Write value 1 to select the sub-menu number 1.
5002h	1	Selezione numero parametro Parameter number selection	4 read – 6 write	Per selezionare il parametro 2 scrivere il valore 2. Write value 2 to select the parameter number 2.
5004h	1	Valore parametro Parameter value	4 read – 6 write	Inserire il valore del parametro. Insert the value of the parameter.
2F03h	1	Salvataggio in memoria Save to flash memory	6 write	Valore=5. Value=5.

ESEMPIO / EXAMPLE
Impostare a 10 secondi il valore del parametro P01.04 (tempo di accelerazione).
Set to 10 seconds the value of the parameter P01.04 (acceleration time).

Passo 1 :Selezione menu 01.
Step 1 :Select menu 01.

MASTER Funzione / Function = 6
Indirizzo / Address = 5000H (5000H – 0001H =4FFFH)
Valore / Value = 1 (01H)

01	06	4F	FF	00	01	6E	EE

ADXL Funzione / Function = 6
Indirizzo / Address = 5000H (5000H – 0001H =4FFFH)
Valore / Value = 1 (01H)

01	06	4F	FF	00	01	6E	EE

(Nota: in questo caso di esempio non è necessaria impostazione del sotto-menu all'indirizzo 5001)
(Note: In this example it is not necessary to set the sub-menu number at address 5001)

Passo 2 :Selezione parametro 04.
Step 2 :Select parameter 04.

MASTER Funzione / Function = 6
Indirizzo / Address = 5002H (5002H – 0001H =5001H)
Valore / Value = 4 (04H)

01	06	50	01	00	04	C8	C9

ADXL Funzione / Function = 6
Indirizzo / Address = 5002H (5002H – 0001H =5001H)
Valore / Value = 4 (04H)

01	06	50	01	00	04	C8	C9

Passo 3 :Impostazione valore 10.
Step 3 :Set value 10.

MASTER Funzione / Function = 6
Indirizzo / Address = 5004H (5004H – 0001H =5003H)
Valore / Value = 10 (0AH)

01	06	50	03	00	0A	E8	CD

ADXL Funzione / Function = 6
Indirizzo / Address = 5004H (5004H – 0001H =5003H)
Valore / Value = 10 (0AH)

01	06	50	03	00	0A	E8	CD

Passo 4: Salvataggio e riavvio.
Step 4: Saving and reboot.

MASTER Funzione / Function = 6 (06H)
Indirizzo / Address = 2F03H (2F03H – 0001H =2F02H)
Valore / Value = 5 (05H)

01	6	2F	02	00	05	E0	DD

ADXL Il dispositivo effettua il salvataggio dei parametri ed esegue il reboot (non si riceve nessuna risposta via Modbus).
The device saves and reboots (no Modbus response message will be received).