



Regolatore elettronico di flusso proporzionale

Serie EV10

Protocollo MODBUS

REVISIONI

| Rev. | Data | Descrizione |
|------|------------|--|
| 0 | 16/03/2022 | Emissione |
| 1 | 22/07/2022 | Aggiunte informazioni comunicazione RS485 |
| 2 | 17/02/2023 | Aggiunte informazioni comando checksum |
| 3 | 27/02/2025 | Aggiunti dettagli impostazione address RS485 |
| | | |
| | | |
| | | |

INDICE

| | |
|--------------------------------------|----|
| 1 INTRODUZIONE..... | 3 |
| 2 PROTOCOLLO MODBUS..... | 3 |
| 2.1 RS485 | 3 |
| 2.2 FORMATTAZIONE RTU STANDARD | 3 |
| 3 PROTOCOLLO EV10..... | 4 |
| 3.1 BOOTLOADER | 4 |
| 3.1.1 KEEP ALIVE BOOT MODE | 4 |
| 3.1.2 COMANDO CHECKSUM | 5 |
| 3.1.3 COMANDI BOOT MODE..... | 5 |
| 3.1.3.1 FLASH ERASE | 5 |
| 3.1.3.2 FLASH WRITE..... | 6 |
| 3.1.3.3 JUMP TO APPLICATION | 6 |
| 3.2 APPLICAZIONE | 6 |
| 3.2.1 BOOTLOADER REQUEST | 6 |
| 3.2.2 SET NODE ID COMMAND | 6 |
| 3.2.3 CALIBRATION | 7 |
| 3.2.4 MAX STEP | 7 |
| 3.2.5 PERCENTAGE COMMAND | 7 |
| 3.2.6 TEMPERATURE..... | 7 |
| 3.2.7 STATUS BOARD | 8 |
| 3.2.8 ERROR LIST | 8 |
| 3.2.9 INPUT COMMAND..... | 9 |
| 3.2.10 SERIAL NUMBER..... | 9 |
| 3.2.11 GET POSITION..... | 9 |
| 3.2.12 FIRMWARE VERSION | 9 |
| 3.3 CRC EXAMPLE FUNCTION..... | 10 |

1 INTRODUZIONE

Questo documento contiene le specifiche relative al protocollo MODBUS implementato nel progetto EV10 di CMATIC S.P.A.

2 PROTOCOLLO MODBUS

Il protocollo MODBUS del regolatore elettronico è di tipo RTU e di default il NODE ID pari a 0x00.

Per rendere operativo il regolatore elettronico e poterlo gestire all'interno di un bus RS485 è necessario configurare lo specifico NODE ID che si vuole utilizzare mediante il comando specificato nella sezione 3.2.2 SET NODE ID COMMAND.

2.1 RS485

L'interfaccia digitale utilizzata per il protocollo MODBUS è la RS485 (2 fili). La configurazione della porta seriale è la seguente:

- Velocità (Baud Rate): 115200
- Bits di dati: 8
- Parità: None
- Stop Bits: 1

2.2 FORMATTAZIONE RTU STANDARD

Il modello di protocollo MODBUS implementato nel progetto EV10 presenta la formattazione dei pacchetti RTU (Hex Addressing).

Ogni pacchetto è composto come segue:

| Nome | Lunghezza (bits) |
|--------------|------------------|
| Indirizzo | 8 |
| Funzione | 8 |
| Data Address | 16 |
| Data | n × 16 |
| CRC | 16 |

Il CRC è di tipo polinomiale $x^{16} + x^{15} + x^2 + 1$, mentre n può valere fino a 5. Il pacchetto di errore invece è costituito come:

| Nome | Lunghezza (bits) |
|-----------|---------------------------|
| Indirizzo | 8 |
| Funzione | 0x80 + Funzione richiesta |
| Data | Codice Errore |
| CRC | 16 |

N.B.: dopo aver ricevuto un pacchetto di risposta è necessario aspettare almeno 10 ms prima di inviare la richiesta successiva.

3 PROTOCOLLO EV10

Il protocollo implementato nel Sistema EV10 è composto da diversi comandi di lettura e scrittura utilizzati per comunicare con il bootloader dell'EV10 e con l'applicativo stesso del Sistema. La scheda EV10 sulla rete 485 è di tipo slave e rimane sempre in attesa di un comando. Ad ogni pacchetto ricevuto, corrisponde o una risposta specifica (in caso di read, altrimenti lo stesso pacchetto in caso di write), oppure un pacchetto di errore.

Pacchetto errore:

| Codice Errore | Nome |
|---------------|----------------------|
| 0x0001 | ILLEGAL FUNCTION |
| 0x0002 | ILLEGAL DATA ADDRESS |
| 0x0003 | ILLEGAL DATA VALUE |
| 0x0004 | SLAVE DEVICE FAILURE |

3.1 BOOTLOADER

3.1.1 KEEP ALIVE BOOT MODE

WRITE ONLY

Funzione = 0x1002

Data = 0x0001

Questo comando permette di mantenere la scheda in bootloader.

Dopo l'invio del comando la scheda rimarrà in BOOT MODE per 15 secondi.

3.1.2 COMANDO CHECKSUM

READ ONLY

Funzione = 0x1001

Questo comando permette di leggere il checksum del FW attualmente caricato sulla scheda. Viene calcolato con la stessa funzione riportata al capitolo 4.3.

Dopo l'invio del comando la scheda rimarrà in BOOT MODE per 15 secondi.

3.1.3 COMANDI BOOT MODE

Questi comandi provengono da un protocollo interno, verranno comunque inviati tramite protocollo Modbus con la scrittura dei registri multipli (comando 0x10), ad indirizzo 0x1000.

Il pacchetto dovrà essere inviato nel campo Data attenendosi alla seguente forma:

se il pacchetto da inviare risulta dispari, dovrà essere aggiunto un byte (0xFF) in coda.

| Nome | Lunghezza (bits) |
|------------------|--------------------------|
| Slave ID | 8 |
| Funzione | 8 (0x10) |
| Indirizzo | 16 |
| Lunghezza dati | 16 |
| Byte da scrivere | 8 = (Lunghezza dati * 2) |
| Data ... 1 | 16 |
| Data ... 2 | 16 |
| Data ... n | 16 |
| CRC | 16 |

3.1.3.1 FLASH ERASE

Questo comando serve a preparare la memoria all'aggiornamento. Deve essere inviato prima di un qualsiasi aggiornamento. Dopo aver spedito il comando, il timer di JUMP APPLICATION verrà resettato a 15 secondi.

| | | | | | | | |
|-------|-------|---------|---------|---------|---------|---------|------|
| 0xFD | 0xDF | 0x00 | 0x01 | 0x03 | CRC | CRC | 0xFF |
| START | START | N. DATI | N. DATI | COMMAND | CRC LSB | CRC MSB | |

3.1.3.2 FLASH WRITE

Questo comando serve a scrivere la memoria flash. L'indirizzo di partenza è 0x2000. L'indirizzo di fine è 0xFFFF.

I file di aggiornamento sono di grandezza (0xFFFF - 0x2000), e devono essere scritti interamente nella memoria interna del micro. La dimensione massima di Byte scritti in un solo pacchetto è di 64.

Dopo aver spedito il comando, il timer di JUMP APPLICATION verrà resettato a 15 secondi.

| | | | | | | |
|-------|-------|---------|---------|---------|--------------|--------------|
| 0xFD | 0xDF | 0xNN | 0xNN | 0x04 | ADDR MSB | ADDR LSB |
| START | START | N. DATI | N. DATI | COMMAND | MEM. ADDRESS | MEM. ADDRESS |

| | | | | | | |
|--------|--------|---------|--------|---------|---------|------|
| DATA 1 | DATA 2 | DATA .. | DATA N | CRC | CRC | 0xFF |
| DATO 1 | DATO 2 | DATO .. | DATO N | CRC LSB | CRC MSB | |

3.1.3.3 JUMP TO APPLICATION

Questo comando serve per passare direttamente all'applicazione senza attendere il tempo di JUMP APPLICATION.

| | | | | | | | |
|-------|-------|---------|---------|---------|---------|---------|------|
| 0xFD | 0xDF | 0x00 | 0x01 | 0x07 | CRC | CRC | 0xFF |
| START | START | N. DATI | N. DATI | COMMAND | CRC LSB | CRC MSB | |

3.2 APPLICAZIONE

Qui di seguito sono indicati i vari comandi disponibili per la parte applicativa

3.2.1 BOOTLOADER REQUEST

WRITE ONLY

Funzione = 0x01

Data = 0x0001

Questo comando permette di riavviare la scheda in bootloader direttamente dall'applicativo.

3.2.2 SET NODE ID COMMAND

WRITE

Data address = 0x02

Data = 0x0001 - 0x00FE

Questo comando permette di impostare un nuovo indirizzo del nodo sul bus di comunicazione RS485.

Nel caso in cui non si conosca l'indirizzo specifico del regolatore elettronico è comunque disponibile un indirizzo NODE ID BROADCAST pari a 0xFF. Questo significa che, collegando uno ed un solo regolatore elettronico sul bus RS485 ed inviando il comando SET NODE ID COMMAND indirizzato al nodo 0xFF, l'unico

regolatore elettronico collegato, indipendentemente dal nodo ID precedentemente configurato, riceverà il comando di set e memorizzerà il nuovo NODE ID.

Dalla successiva accensione riceverà comandi al solo NODE ID precedentemente configurato. A questo punto il regolatore elettronico può essere reinserito all'interno del bus RS485 e gestito insieme agli altri regolatori collegati a tutto l'impianto.

3.2.3 CALIBRATION

WRITE/READ

Data address = 0x03

Data = 0x0000 CALIB_READY

0x0001 CALIB_START

0x0002 CALIB_WAIT_1

0x0003 CALIB_RUN_HOME

0x0004 CALIB_WAIT_2

0x0005 CALIB_RUN_CLOSE

0x0006 CALIB_END

0x0007 CALIB_ERROR

Questo comando consente di avviare e leggere lo stato della calibrazione del motore. L'unico comando che è possibile scrivere è il CALIB_START se lo stato attuale è CALIB_READY e il motore è attualmente fermo.

3.2.4 MAX STEP

READ ONLY

Data address = 0x04 (LSB), 0x05 (MSB)

Data = 0x0000 - 0xFFFF

Questo comando permette di leggere il valore degli step calcolati durante la calibrazione.

3.2.5 PERCENTAGE COMMAND

WRITE/READ

Data address = 0x06

Data = 0x0000 - 0x0064

Questo comando permette di impostare la percentuale di apertura della valvola collegata alla scheda EV10.

3.2.6 TEMPERATURE

READ ONLY

Data address = 0x07

Data = 0x0000 - 0x04E2

Questo comando permette di leggere il valore della temperatura rilevata sulla scheda EV10.

Esempio:

Data = 0x0160 ⇒ 352 ⇒ 35,2 °C

3.2.7 STATUS BOARD

READ ONLY

Data address = 0x08

Data = 0x0000 BOARD_OFF

0x0001 BOARD_READY

0x0002 BOARD_MOTOR_RUNNING

0x0003 BOARD_ERROR

0x0004 BOARD_MOTOR_RUNNING_WITH_ERROR

0x0005 BOARD_CALIBRATION

Questo comando permette di leggere lo stato attuale dell'applicazione della scheda EV10.

3.2.8 ERROR LIST

WRITE/READ

Data address = 0x09

| | |
|---------------------------------|----------------|
| Data = NO_ERROR | 0x0000 |
| FIRST_HOMING_ERROR | (0x0001 << 0) |
| STALL_GUARD_ERROR | (0x0001 << 1) |
| SHORT_LOW_SIDE_CIRCTUIT_PHASE_A | (0x0001 << 2) |
| SHORT_LOW_SIDE_CIRCTUIT_PHASE_B | (0x0001 << 3) |
| SHORT_GND_CIRCTUIT_PHASE_A | (0x0001 << 4) |
| SHORT_GND_CIRCTUIT_PHASE_B | (0x0001 << 5) |
| OVERTEMP_PRE_WARNING | (0x0001 << 6) |
| OVERTEMP_DETECTED | (0x0001 << 7) |
| CALIBRATION_ERROR | (0x0001 << 8) |
| TIMEOUT_ERROR | (0x0001 << 9) |
| MOTOR_CONTROL_ERROR | (0x0001 << 10) |

Questo comando consente di avviare e leggere gli errori attuali dell'applicazione della scheda EV10.

Nel caso di scrittura, il bit settato nel pacchetto inviato verso la EV10 verrà resettato.

Esempio:

errori attuali 0x0003 ⇒ Inviato 0x0001 errore ⇒ attuali 0x0002

3.2.9 INPUT COMMAND

WRITE/READ

Data address = 0x0A

Data = 0x0000 (ingresso analogico) - 0x0001 (RS485 - MODBUS).

Questo comando permette di impostare l'ingresso di comando della scheda EV10.

3.2.10 SERIAL NUMBER

WRITE/READ

Data address = 0x0B (LSB), 0x0C, 0x0D, 0x0E, 0x0F (MSB)

Data = 0x0000 - 0xFFFF

Questo comando permette di impostare il numero seriale della scheda EV10.

Esempio:

comandi 0x3132, 0x3334, 0x3536, 0x3738, 0x3900

Seriale: 123456789

3.2.11 GET POSITION

READ ONLY

Data address = 0x10

Data = 0x0000 - 0x0064

Questo comando permette di leggere l'attuale posizione in percentuale del motore comandato dalla EV10.

3.2.12 FIRMWARE VERSION

READ ONLY

Data address = 0x11 (Major), 0x12 (Minor)

Data = 0x0000 - 0xFFFF

Questo comando permette di leggere la versione dell'applicativo attualmente carica sulla scheda EV10.

Esempio:

ricevuto 0x0001, 0x0002

Versione Firmware: 01.02

3.3 CRC EXAMPLE FUNCTION

```
long crc16_MODBUS(int lg_buffer, unsigned char *pt_buffer)
{
    int no_octet;
    unsigned char val_octet;
    int no_bit;
    int retenue;
    long crc;
    crc = 0X0000FFFF;
    for (no_octet=0 ; no_octet < lg_buffer ; no_octet++)
    {
        val_octet = pt_buffer[no_octet];
        crc = crc ^ val_octet;
        for (no_bit=0 ; no_bit < 8 ; no_bit++)
        {
            retenue = crc & 0X0001;
            crc = crc >> 1;
            if (retenue == 1)
                crc = crc ^ 0XA001;
        }
    }
    return(crc);
}
```