Alpha-Log

Protocolli di comunicazione

Documento Alpha-Log – Protocolli di comunicazione
Pagine 29

Lista delle revisioni

Esponente di revisione	***Data***	***Descrizione delle modifiche***
Origine	27/11/2017	
a	12/04/2021	Rimozione Pluvi-ONE; aggiunto AlphaLog e nuove funzionalità di Modbus RTU e parti Modbus TCP
b	15/11/2021	Aggiunta protocollo MQTT; formattazione di alcune tabelle
c	07/03/2022	Aggiunto esempi d'uso con programma modpoll

Note su questo manuale

Le informazioni contenute nel presente manuale sono soggette a modifiche senza preavviso. Nessuna parte di questo manuale può essere riprodotta in qualsiasi forma o mezzo elettronico o meccanico, per alcun uso, senza il permesso scritto di LSI LASTEM.

LSI LASTEM si riserva il diritto di intervenire sul prodotto, senza l'obbligo di aggiornare tempestivamente questo documento.

Sommario

1 Introduzione

Alpha-Log supporta diversi protocolli di comunicazione. Oltre al protocollo nativo *SAP*, supporta Modbus RTU e TCP, SDI-12, MQTT, TCP/IP ed altri ancora.
In questo manuale vengono descritti i protocolli *Modbus RTU, Modbus TCP* nella modalità slave e *MQTT*.

2 Protocollo Modbus RTU

Modbus RTU è un protocollo di comunicazione seriale molto utilizzato in ambito industriale per consentire la comunicazione tra un *master* (solitamente un PC o un sistema SCADA) ed uno o più *slave* (strumentazioni di misura, di controllo o PLC), connessi al medesimo bus di comunicazione. Modbus definisce come i dispositivi *master* e *slave* instaurano ed interrompono la comunicazione, come vengono scambiati i messaggi e come vengono rilevati gli errori. Solo il *master* può iniziare la comunicazione.

Ad ogni dispositivo connesso al bus è assegnato un indirizzo univoco. Un comando Modbus contiene l'indirizzo del dispositivo destinatario verso il quale è diretto il messaggio. Solo il dispositivo indirizzato risponderà al comando, sebbene anche gli altri strumenti ricevano il medesimo messaggio. Tutti i messaggi Modbus trasmessi contengono informazioni di controllo che consentono alla parte ricevente di valutarne la corretta ricezione.

2.1 Formato dei messaggi

Il formato dei messaggi utilizzato dai dispositivi master/slave è il seguente:

Nome campo	*Dimensione*	*Descrizione*
Indirizzo	1 byte	L'indirizzo del dispositivo slave con cui master deve comunicare (indirizzo di rete dello strumento); non è supportato il messaggio *broadcast* (ID = 0)
Codice funzione	1 byte	Comando da eseguire (o eseguito)
Dati	n byte	Campo dati (payload) di lunghezza variabile
CRC16	2 byte	Codice di controllo del messaggio calcolato secondo l'algoritmo CRC16

Dopo la ricezione del messaggio, l'apparato slave esegue l'analisi della corrispondenza fra il proprio indirizzo di rete e quello contenuto nel messaggio; inoltre valuta il codice di controllo rispetto a quello calcolato autonomamente. Se entrambe le condizioni sono verificate, l'apparato slave risponde al messaggio ricevuto, altrimenti scarta il messaggio senza trasmettere alcuna risposta. In caso di ricezione di messaggio in cui la parte dato contenga parametri non corretti, l'apparato slave risponde indicando la condizione di errore tramite un'eccezione (vedi §2.3.4).

2.1.1 Indirizzo di rete

L'indirizzo di rete è utilizzato per identificare il destinatario del messaggio: esso contiene l'indirizzo numerico del dispositivo slave selezionato. Può assumere valori da 1 a 247. Utilizzare il programma *3DOM* è modificare il parametro *Indirizzo* in base a necessità.

I messaggi di tipo *broadcast* (indirizzo uguale a 0) non sono supportati.

2.1.2 Codice funzione

Il codice funzione identifica il comando che deve essere eseguito o che è appena stato gestito dallo slave.
I codici funzione supportati dallo strumento sono i seguenti:

Codice funzione	*Nome funzione*	*Descrizione*
01	Read Coils	Legge lo stato degli attuatori e lo stato di funzionamento dello strumento
03	Read Holding Registers	Legge gli ultimi valori acquisiti dallo strumento (sia misure analogiche che digitali) e la data ora di sistema
04	Read Input Registers	Come *Read Holding Registers*
05	Write Single Coil	Imposta lo stato di un attuatore dello strumento
0F	Write Multiple Coils	Azzera gli errori di funzionamento dello strumento
10	Write Multiple Registers	Imposta la data ora di sistema e alcuni parametri di configurazione delle misure
2B	Read Device Identification	Legge le informazioni dello strumento

Qualsiasi altro codice funzione non inserito in tabella viene ignorato dallo strumento, il quale a fronte della ricezione di un comando non supportato non genera alcuna risposta di errore (eccezione).

2.1.3 CRC16

Il CRC16 contiene il codice di ridondanza ciclica (*Cyclic Redundancy Check*) calcolato utilizzando l'algoritmo CRC16 (polinomio X^16 + X^15 + X^2 + 1). Per il calcolo di questi due caratteri il messaggio (indirizzo, codice funzione e dati scartando i bit di start, stop e l'eventuale parità) viene considerato come un unico numero binario continuo di cui il bit più significativo (MSB) viene trasmesso prima.

2.2 Linea di comunicazione

Alpha-Log supporta Modbus RTU sulle linee di comunicazione RS485 (denominata *Com3*) e RS232 (denominata *Com2*). Fare riferimento al manuale dello strumento per individuare lo schema di connessione elettrica.

2.3 Funzioni supportate

2.3.1 Read Coils

Utilizzare la funzione *Read Coils* per leggere lo stato degli attuatori dello strumento ed eventuali errori di funzionamento.

Nota: il comando non fornisce lo stato degli attuatori disponibili sulle unità *ALIEM* eventualmente connesse ad AlphaLog.

Il significato dei valori assunti è il seguente:

Coils	*Valore*	*Significato*
Stato attuatori	0	Uscita attuazione spenta (OFF)
	1	Uscita attuazione accesa (ON)
Stato di funzionamento	0	Non in errore
	1	Errore

Indirizzo (Hex)	*Coil*	*Significato*
Stato Attuatori		
0x00	01	Stato attuatore nr. 1
	02	Stato attuatore nr. 2
	03	Stato attuatore nr. 3
	04	Stato attuatore nr. 4
	05	*Non utilizzato*
	06	*Non utilizzato*
	07	*Non utilizzato*
	08	*Non utilizzato*
Stato funzionamento (errori)		
0x01	09	Errore Read After Write
	10	Configurazione memorizzata non valida
	11	Errore di ricerca della pagina di memoria
	12	Nr. massimo di loop superato durante ricerca dati in memoria
	13	Overrun coda di accettazione richieste di acquisizione
	14	Overrun coda di memorizzazione risultati di acquisizione
	15	*Non utilizzato*
	16	Scrittura di pagina in memoria fallita
0x02	17	Una o più pagine di dati di elaborazione perse
	18	Cancellazione settore di memoria fallita
	19	Timeout su attesa termine operazione in memoria
	20	Dispositivo di memoria non supportato
	21	Messaggio ricevuto errato (codice di controllo non valido)
	22	Messaggio ripetuto fino ad un massimo di 3 volte
	23	A seguito della ripetizione massima del messaggio questo è stato perso
	24	Errore riscontrato durante la scrittura/lettura della EEProm interna allo slave
0x03	25	Lo slave non è in grado di operare in quanto la sua config. non è programmata
	26	C'è stato un overflow sulla ricezione di un singolo messaggio
	27	Coda dei messaggi piena nello slave
	28	Raggruppa tutti gli errori CISS relativi alla sintassi
	29	Errore CISS non specificato (Unspecified)
	30	Errore CISS (BadCommandCode)
	31	Errore CISS (BadParameter)
	32	Errore CISS (ParameterOutOfRange)
0x04	33	Errore CISS (UnrecognizedCDV)
	34	Errore CISS (BeyondMaxClassLevel)
	35	Errore CISS (ParameterIndexOutOfRange)
	36	Errore CISS (ClassIndexOutOfRange)
	37	Errore CISS (RequestNotPermitted)
	38	*Non utilizzato*
	39	*Non utilizzato*
	40	*Non utilizzato*

Richiesta

Codice funzione	1 byte	0x01
Indirizzo inizio	2 byte	da 0x00 a 0x27
Numero di coils	2 byte	da 1 a 40

Risposta

Codice funzione	1 byte	0x01
Numero di byte	1 byte	1
Stato attuatore/errore	1 byte	1=On, 0=Off

Errore

Codice funzione	1 byte	0x81
Codice eccezione	1 byte	01 o 02 o 03 o 04

Per maggiori informazioni su *Codice eccezione* fare riferimento al §2.4.

Esempio: richiesta di lettura dello stato degli attuatori dello strumento con ID uguale a 01:

Richiesta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	01
Indirizzo inizio (Hi)	00
Indirizzo inizio (Lo)	00
Numero di attuatori (Hi)	00
Numero di attuatori (Lo)	08
CRC16 (Hi)	3D
CRC16 (Lo)	CC

Risposta alla richiesta con i seguenti valori: 0, 0, 1, 0, 0, 0, 0, 0:

Risposta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	01
Numero di byte	01
Valore	04
CRC16 (Hi)	50
CRC16 (Lo)	4B

2.3.2 Read Holding Registers

Read Holding Registers e *Read Input Registers*, dal punto di vista dell'implementazione Modbus nello strumento, si equivalgono e sono perciò trattati nello stesso modo. Nella richiesta indicare il codice funzione 0x03 (*Read Holding Registers*) oppure 0x04 (*Read Input Registers*).

Per maggiori informazioni in merito fare riferimento alla funzione *Read Input Registers.*

2.3.3 Read Input Registers

La funzione *Read Input Registers* permette di leggere gli ultimi valori acquisiti dallo strumento (misure istantanee campionate o calcolate) e la data ora di sistema. I valori acquisiti possono essere letti in formato a virgola mobile richiedendoli dall'indirizzo 0 (0x0000) ed in formato binario intero dall'indirizzo 1000 (0x03E8), mentre la data ora è disponibile all'indirizzo 2000 (0x07D0).

L'indirizzo da impostare per ciascuna richiesta è il seguente:

Indirizzo (Hex)	*Numero di registri*	*Significato*
Valori float IEEE754		
0x0000	2	Valore della misura 1
0x0002	2	Valore della misura 2
…	…	…
0x00C4	2	Valore della misura 99
Valori interi (WORD)		
0x03E8	1	Valore della misura 1
0x03E9	1	Valore della misura 2
…	…	…
0x044B	1	Valore della misura 99
Data ora di sistema (yy MM dd hh mm ss)		
0x07D0	1	yy MM
0x07D1	1	dd hh
0x07D2	1	mm ss

2.3.3.1 *Valori espressi in formato a virgola mobile*

Ogni misura tramessa in virgola mobile è composta da 4 byte ovvero da 2 registri Modbus. Essi sono espressi in base al formato floating point indicato dalla norma IEEE754.

Tramite il programma 3DOM è possibile scegliere il formato determinandone l'inversione dei byte che lo costituiscono (opzione *Inversione valore a virgola mobile* disponibile nella maschera di impostazione dei parametri di sistema). La trasmissione del dato, con opzione impostata a *false*, inizia dal byte più significativo (MSB – Most Significant Byte) mentre con opzione impostata a *true* inizia dal byte meno significativo (LSB – Least Significant Byte). Il valore 11,0 viene quindi memorizzato, a partire dall'indirizzo 0x20, ad esempio, come segue:

			Indirizzo di memoria (Hex)	**Senza inversione del valore**	**Con inversione del valore**
			…		
Valore 11,0	Registro 1	Byte 1	0x20 (Hi)	00	30
		Byte 2	0x20 (Lo)	00	41
	Registro 2	Byte 1	0x21 (Hi)	41	00
		Byte 2	0x21 (Lo)	30	00
			…		

Essendo il frame composto da 255 caratteri, è possibile richiedere al massimo 60 misure per ogni messaggio di richiesta. Per richiedere tutte le 99 misure potenzialmente disponibili nello strumento si rende quindi necessario effettuare due richieste distinte.
Il valore -999999 (0xF02374C9), se non diversamente specificato nella configurazione dello strumento (parametro impostabile tramite 3DOM), corrisponde a *misura in errore*.

Richiesta

Codice funzione	1 byte	0x04
Indirizzo inizio	2 byte	da 0x0000 a 0x00C4
Numero di registri	2 byte	da 2 a 198 (max 120)

Risposta

Codice funzione	1 byte	0x04
Numero di byte	1 byte	2 x N*
Valore	2 byte x N*	

*N = numero di misure.

Errore

Codice funzione	1 byte	0x84
Codice eccezione	1 byte	01 o 02 o 03 o 04

Per maggiori informazioni su *Codice eccezione* fare riferimento al cap. 2.4.

Area dati

Nr. Misura	1	2	3	4	5	6	…	99
Indirizzo (hex)	0x00	0x02	0x04	0x06	0x08	0x0A	…	0x0C4

Esempio: richiesta di lettura dei valori, nel formato a virgola mobile, delle misure 3 e 4 dello strumento con indirizzo di rete uguale a 01:

La prima misura da leggere è la numero 3 quindi, come indicato nell'*Area dati*, l'indirizzo di inizio da impostare è 0x04 mentre il numero di uscite (registri) è 0x04 (2 misure per 2 registri).

Richiesta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	04
Indirizzo inizio (Hi)	00
Indirizzo inizio (Lo)	04
Numero di uscite (Hi)	00
Numero di uscite (Lo)	04
CRC16 (Hi)	B0
CRC16 (Lo)	08

Risposta alla richiesta con i valori 99.0 per la misura 3 (valore uscita 1) e 101.0 per la misura 4 (valore uscita 2):

Risposta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	04
Numero di byte	08
Valore uscita 1 (byte 1)	00
Valore uscita 1 (byte 2)	00
Valore uscita 1 (byte 3)	42
Valore uscita 1 (byte 4)	C6
Valore uscita 2 (byte 1)	00
Valore uscita 2 (byte 2)	00
Valore uscita 2 (byte 3)	42
Valore uscita 2 (byte 4)	C4
CRC16 (Hi)	13
CRC16 (Lo)	C9

2.3.3.2 *Valori espressi in formato binario intero*

Nel formato binario ogni misura è di tipo intero, quindi composta da 2 byte ovvero da un registro Modbus. Il formato dei dati è *Little Endian*. La memorizzazione del dato inizia dal byte meno significativo (LSB – Least Significant Byte). Il valore 1149 (0x0545) viene quindi memorizzato, a partire dall'indirizzo 0x03E8, ad esempio, come segue:

			Indirizzo di memoria (Hex)	**Valore in formato Little Endian (Hex)**
			…	
Valore 1149	Registro 1	Byte 1	0x03E8 (Hi)	05
		Byte 2	0x03E8 (Lo)	45

Lo strumento utilizza l'impostazione della precisione di ogni specifica misura (numero di cifre decimali) per calcolare il valore intero trasmetto da Modbus. Per esempio, il valore 23,45 di una misura impostata con due valori decimali diventa 2345, mentre il valore 68,4 di una misura impostata con un valore decimale diventa 684.

Al contrario del formato a virgola mobile, per il tipo intero è possibile ottenere tutte le 99 misure potenzialmente gestibili da Alpha-Log tramite una singola richiesta.
Il valore -1 (0xFFFF), se non diversamente specificato nella configurazione dello strumento (parametro impostabile tramite 3DOM), corrisponde a *misura in errore*.

Richiesta

Codice funzione	1 byte	0x04
Indirizzo inizio	2 byte	da 0x03E8 a 0x044B
Numero di registri	2 byte	da 1 a 99

Risposta

Codice funzione	1 byte	0x04
Numero di byte	1 byte	2 x N*
Valore	2 byte x N*	

*N = numero di misure.

Errore

Codice funzione	1 byte	0x84
Codice eccezione	1 byte	01 o 02 o 03 o 04

Per maggiori informazioni su *Codice eccezione* fare riferimento al cap. 2.4.

Area dati

Nr. Misura	1	2	3	4	…	99
Indirizzo (hex)	0x03E8	0x03E9	0x03EA	0x03EB	…	0x044B

Esempio: richiesta di lettura dei valori, nel formato intero, della misura 3 dello strumento con ID uguale a 01:

La misura da leggere è la numero 3 quindi, come indicato nell'*Area dati*, l'indirizzo di inizio da impostare è 0x03EA mentre il numero di uscite (registri) è 0x01 (1 misura per 1 registro).

Richiesta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	04
Indirizzo inizio (Hi)	03
Indirizzo inizio (Lo)	EA
Numero di uscite (Hi)	00
Numero di uscite (Lo)	01
CRC16 (Hi)	A5
CRC16 (Lo)	BA

Risposta alla richiesta con il valore 1343

Risposta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	04
Numero di byte	02
Valore uscita 1 (byte 1)	05
Valore uscita 1 (byte 2)	3F
CRC16 (Hi)	FB
CRC16 (Lo)	04

2.3.3.3 *Data ora dello strumento*

Richiesta

Codice funzione	1 byte	0x04
Indirizzo inizio	2 byte	da 0x07D0 a 0x07D2
Numero di registri	2 byte	da 1 a 3

Risposta

Codice funzione	1 byte	0x04
Numero di byte	1 byte	2 x N*
Valore	2 byte x N*	

*N = numero di registri.

La sequenza dei campi data/ora a partire dall'indirizzo 0x07D0 è: yy MM dd hh mm ss.

Errore

Codice funzione	1 byte	0x84
Codice eccezione	1 byte	01 o 02 o 03 o 04

Per maggiori informazioni su *Codice eccezione* fare riferimento al cap. 2.4.

Esempio: richiesta di lettura della data ora dello strumento con ID uguale a 01:

Richiesta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	04
Indirizzo inizio (Hi)	07
Indirizzo inizio (Lo)	D0
Numero di uscite (Hi)	00
Numero di uscite (Lo)	03
CRC16 (Hi)	B0
CRC16 (Lo)	86

Risposta alla richiesta con data ora uguale a 01/02/21 10:36:42

Risposta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	04
Numero di byte	06
Valore uscita 1 (byte 1)	15
Valore uscita 1 (byte 2)	02
Valore uscita 2 (byte 1)	01
Valore uscita 2 (byte 2)	0A
Valore uscita 3 (byte 1)	24
Valore uscita 3 (byte 2)	2A
CRC16 (Hi)	A4
CRC16 (Lo)	D7

2.3.4 Write Single Coil

La funzione *Write Single Coil* consente di modificare lo stato degli attuatori dello strumento (uscite digitali).

Il valore *0x0000* imposta l'uscita dell'attuatore a *0* mentre *0xFF00* la imposta ad *1*. Lo stato *0* indica attuatore spento ed *1* attuatore acceso.
Nota: il comando non consente la modifica dello stato degli attuatori disponibili sulle unità *ALIEM* eventualmente connesse ad AlphaLog.

Richiesta

Codice funzione	1 byte	0x05
Indirizzo inizio	2 byte	da 0x0000 a 0x0003
Valore	2 byte	0x0000 o 0xFF00

Risposta

Codice funzione	1 byte	0x01
Indirizzo inizio	2 byte	da 0x0000 a 0x0007
Valore	2 byte	0x0000 o 0xFF00

Errore

Codice funzione	1 byte	0x85
Codice eccezione	1 byte	01 o 02 o 03 o 04

Per maggiori informazioni su *Codice eccezione* fare riferimento al cap. 2.4.

Esempio: richiesta di impostare a 0 lo stato dell'attuatore numero 3 dello strumento con ID uguale a 01:

Richiesta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	05
Indirizzo inizio (Hi)	00
Indirizzo inizio (Lo)	02
Valore (Hi)	00
Valore (Lo)	00
CRC16 (Hi)	6C
CRC16 (Lo)	0A

Risposta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	05
Indirizzo inizio (Hi)	00
Indirizzo inizio (Lo)	02
Valore (Hi)	00
Valore (Lo)	00
CRC16 (Hi)	6C
CRC16 (Lo)	0A

2.3.5 Write Multiple Coils

Utilizzare la funzione *Write Multiple Coils* per azzerare eventuali errori di funzionamento riscontrati con la funzione *Read Coils*. Il comando agisce su tutti i bit insieme; non è possibile azzerare uno o una sola parte di errori. Il comando, come descritto di seguito, azzera tutti gli errori verificatisi.

Richiesta

Codice funzione	1 byte	0x0F
Indirizzo inizio	2 byte	0x0000
Numero di coil	2 byte	32
Numero di byte	1 byte	4
Valore	4 byte	0x0000

Risposta

Codice funzione	1 byte	0x0F
Indirizzo inizio	2 byte	0x0000
Numero di coil	2 byte	32

Errore

Codice funzione	1 byte	0x8F
Codice eccezione	1 byte	01 o 02 o 03 o 04

Per maggiori informazioni su *Codice eccezione* fare riferimento al cap. 2.4.

Esempio: richiesta di azzeramento degli errori di funzionamento dello strumento con ID uguale a 01:

Richiesta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	0F
Indirizzo inizio (Hi)	00
Indirizzo inizio (Lo)	00
Numero di coil (Hi)	00
Numero di coil (Lo)	20
Numero di byte	04
Valore 1 (Hi)	00
Valore 1 (Lo)	00
Valore 2 (Hi)	00
Valore 2 (Lo)	00
CRC16 (Hi)	C4
CRC16 (Lo)	88

Risposta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	0F
Indirizzo inizio (Hi)	00
Indirizzo inizio (Lo)	00
Numero di coil (Hi)	00
Numero di coil (Lo)	20
CRC16 (Hi)	21
CRC16 (Lo)	79

2.3.6 Write Multiple Registers

La funzione *Write Multiple Registers* permette di impostare la data ora dello strumento e alcuni parametri di configurazione delle misure.

L'indirizzo da utilizzare per ciascuna richiesta è il seguente:

Indirizzo (Hex)	***Numero di registri***	***Significato***
Data ora di sistema (yy mm dd hh mm ss)		
0x07D0	3	yy mm dd hh mm ss
Configurazione misure		
0x07DA	5	Valore assunto per misura in errore per tipo decimale binario, floating point e attivazione misura (fino max 32)

I parametri di configurazione delle misure che possono essere modificati sono:

- Valore da attribuire alla misura in errore per misure trasmesse in formato WORD (2 byte = 1 registro);
- Valore da attribuire alla misura in errore per misure trasmesse in formato float (4 byte = 2 registri); ATTENZIONE: utilizzare il formato *Little Endian;*
- Attivazione della misura (un bit per ogni stato di attivazione della misura per un massimo di 32 misure; 4 byte = 2 registri); la misura non attivata verrà considerata in errore.

Questi parametri sono memorizzati in modo definitivo fino alla loro prossima reimpostazione o fino alla riconfigurazione dello strumento da parte di 3DOM. In questo caso essi sono impostati con valori di default: -1 per i valori decimale binario, -999999 per i valori in virgola mobile, tutte le misure abilitate.
La scrittura deve essere eseguita su tutti i 5 registri per intero.

Esempio 1: richiesta di modifica della data ora dello strumento con il valore 09/06/21 16:03:05 (formato yy/mm/dd hh:mm:ss):

Richiesta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	10
Indirizzo inizio (Hi)	07
Indirizzo inizio (Lo)	D0
Numero di registri (Hi)	00
Numero di registri (Lo)	03
Numero di byte	06
Valore 1 (Hi)	15
Valore 1 (Lo)	06
Valore 2 (Hi)	09
Valore 2 (Lo)	10
Valore 3 (Hi)	03
Valore 3 (Lo)	05
CRC16 (Hi)	B0
CRC16 (Lo)	32

Risposta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	10
Indirizzo inizio (Hi)	07
Indirizzo inizio (Lo)	D0
Numero di coil (Hi)	00
Numero di coil (Lo)	03
CRC16 (Hi)	80
CRC16 (Lo)	85

Esempio 2: richiesta di modifica dei parametri delle misure con i seguenti valori: -12345 (0xCFC7) per valore in errore per il tipo decimale binario, -12345678 (0x4E613CCB) per valore in errore per il tipo float e disattivazione di tutte le misure ad eccezione delle prime 3:

Richiesta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	10
Indirizzo inizio (Hi)	07
Indirizzo inizio (Lo)	DA
Numero di registri (Hi)	00
Numero di registri (Lo)	05
Numero di byte	0A
Valore 1 (Hi)	C7
Valore 1 (Lo)	CF
Valore 2 (Hi)	4E
Valore 2 (Lo)	61
Valore 3 (Hi)	3C
Valore 3 (Lo)	CB
Valore 4 (Hi)	07
Valore 4 (Lo)	00
Valore 5 (Hi)	00
Valore 5 (Lo)	00
CRC16 (Hi)	6C
CRC16 (Lo)	11

Risposta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	10
Indirizzo inizio (Hi)	07
Indirizzo inizio (Lo)	DA
Numero di coil (Hi)	00
Numero di coil (Lo)	05
CRC16 (Hi)	20
CRC16 (Lo)	85

2.3.7 Read Device Identification

La funzione *Read Device Identification* consente di ottenere informazioni sullo strumento quali: nome dell'azienda produttrice, tipo, codice, numero di serie e versione dello strumento.

Richiesta

Codice funzione	1 byte	0x2B
Tipo MEI	1 byte	0x0E
Codice Id lettura dispositivo	1 byte	01
Id oggetto	1 byte	0x00

Risposta

Codice funzione	1 byte	0x2B
Tipo MEI	1 byte	0x0E
Codice Id lettura dispositivo	1 byte	01
Livello di conformità	1 byte	0x01
Segue	1 byte	0
Id oggetto successivo	1 byte	Numero Id dell'oggetto
Numero di oggetti	1 byte	3
Id oggetto 1	1 byte	0
Lunghezza oggetto 1	1 byte	26
Valore oggetto 1	Lunghezza oggetto 1	"LSI-Lastem - Milan (Italy)"
Id oggetto 2	1 byte	1
Lunghezza oggetto 2	1 byte	23
Valore oggetto 2	Lunghezza oggetto 2	Tipo, codice, numero di serie di fabbrica e nome definito dall'utente
Id oggetto 3	1 byte	2
Lunghezza oggetto 3	1 byte	7
Valore oggetto 3	Lunghezza oggetto 3	Versione dello strumento

Errore

Codice funzione	1 byte	Codice funzione + 0x80
Codice eccezione	1 byte	01 o 02 o 03 o 04

Per maggiori informazioni su *Codice eccezione* fare riferimento al cap. 2.4.

Esempio: richiesta di lettura dei dati identificativi di Alpha-Log di tipo ALP-001, numero di serie 18020267, versione firmware 2.01.00 con ID impostato a 01:

Richiesta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	2B
Tipo MEI	0E
Codice Id lettura dispositivo	01
Id oggetto	00
CRC16 (Hi)	70
CRC16 (Lo)	77

Risposta	
Nome campo	***(Hex)***
Indirizzo dispositivo	01
Codice funzione	2B
Tipo MEI	0E
Codice Id lettura dispositivo	01
Livello di conformità	01
Segue	00
Id oggetto successivo	00
Numero di oggetti	03
Id oggetto 1	00
Lunghezza oggetto 1	1A
Valore oggetto 1*	“LSI-Lastem - Milan (Italy)”
Id oggetto 2	01
Lunghezza oggetto 2	17
Valore oggetto 2*	“ALP-001; Serial18020267”
Id oggetto 3	02
Lunghezza oggetto 3	07
Valore oggetto 3*	“2.01.00”
CRC16 (Hi)	27
CRC16 (Lo)	3B

*Valori in esadecimale per:

- *Valore oggetto 1:* [4C][53][49][2D][4C][61][73][74][65][6D][20][2D][20][4D][69][6C][61][6E][20][28][49][74][61][6C][79][29]
- *Valore oggetto 2:* [41][4C][50][2D][30][30][31][3B][20][53][65][72][69][61][6C][31][38][30][32][30][32][36][37]
- *Valore oggetto 3:* [32][2E][30][31][2E][30][30]

2.4 Codici eccezione

I codici di eccezione sono trasmessi quando il comando inviato all’unità slave, sebbene corretto nella forma, non può essere eseguito. I codici di eccezione restituiti sono i seguenti:

Codice	***Nome***	***Significato***
01	ILLEGAL FUNCTION (non supportato in questo momento)	Il codice di funzione non corrisponde ad una funzione supportata dal dispositivo slave
02	ILLEGAL DATA ADDRESS	Il registro specificato ha indirizzo non valido
03	ILLEGAL DATA VALUE	Il valore da assegnare non è valido per l’indirizzo specificato
04	SLAVE DEVICE FAILURE	Si è verificato un errore durante l’esecuzione del comando richiesto

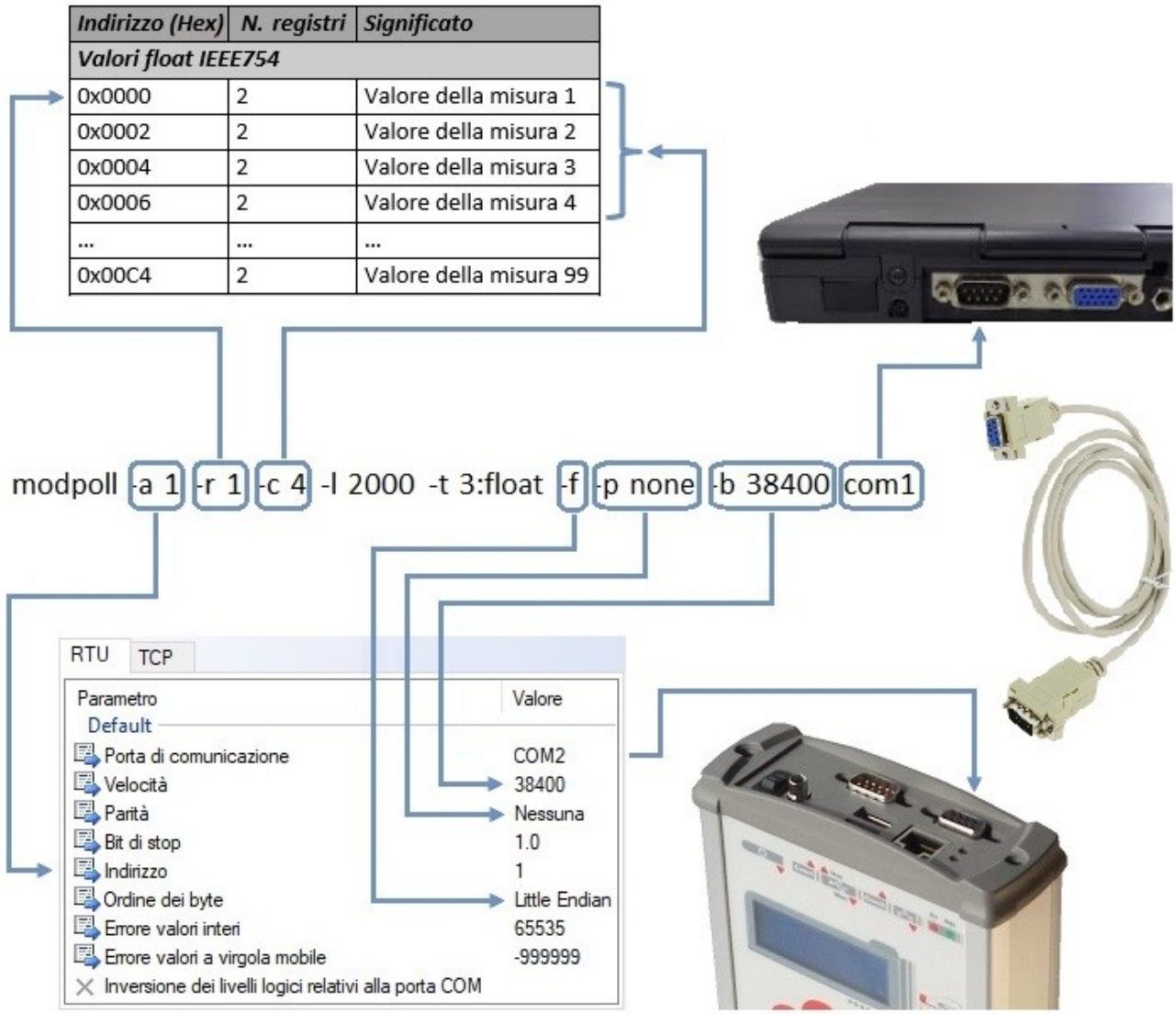
2.5 Verifica con programma modpoll

I codici funzione 0x03 (*Read Holding Registers*) e 0x04 (*Read Input Registers*) possono essere verificati con modpoll.

Modpoll è un programma di simulazione Modbus master basato su riga di comando. È scaricabile gratuitamente dal sito https://www.modbusdriver.com/modpoll.html.

Con il comando *modpoll -h* si ottiene la lista dei parametri che è possibile usare.

A titolo di esempio si riporta la "costruzione" del comando *Read Input Register* per richiedere i valori delle prime 4 misure di Alpha-Log configurato in modalità Modbus Slave sulla seriale Com2.



Di seguito il risultato:

```
modpoll 3.1 - FieldTalk(tm) Modbus(R) Master Simulator
Copyright (c) 2002-2011 proconX Pty Ltd
Visit http://www.modbusdriver.com for Modbus libraries and tools.

Protocol configuration: Modbus RTU
Slave configuration...: address = 1, start reference = 1, count = 4
Communication.....: com1, 38400, 8, 1, none, t/o 1.00 s, poll rate 2000 ms
Data type.....: 32-bit float, input register table
Word swapping.....: Slave configured as big-endian float machine

-- Polling slave... (Ctrl-C to stop)
[1]: 22.604630
[3]: 12.187080
[5]: 1002.520020
[7]: 1002.234985
-- Polling slave... (Ctrl-C to stop)
[1]: 22.608200
[3]: 12.187080
[5]: 1003.099976
[7]: 1002.523254
-- Polling slave... (Ctrl-C to stop)
[1]: 22.608200
[3]: 12.187080
[5]: 1003.099976
[7]: 1002.523254
-- Polling slave... (Ctrl-C to stop)
[1]: 22.608200
[3]: 12.187080
[5]: 1003.099976
[7]: 1002.523254
…
```

3 Protocollo Modbus TCP

Modbus TCP è un'estensione del protocollo Modbus RTU. Esso utilizza l'interfaccia TCP in una rete Ethernet. Il controllo dell'errore non è eseguito tramite CRC ma a livello di frame IP. L'indirizzamento dei dispositivi è gestito tramite IP.

3.1 Porta di comunicazione

Alpha-Log è raggiungibile via Modbus TCP se connesso alla rete tramite la porta Ethernet oppure tramite una chiavetta USB Wi-Fi. Per maggiori informazioni fare riferimento al manuale dello strumento.

3.2 Indirizzamento

L'indirizzamento dei dispositivi è basato su IP. Master e slave, o meglio, Client e Server, per comunicare tra loro devono avere lo stesso range di indirizzi IP.
Utilizzare il programma *3DOM* per impostare l'indirizzo IP.

I messaggi di tipo *broadcast* (indirizzo uguale a 0) non sono supportati.

3.3 Funzioni supportate

Modbus TCP supporta le funzioni *Read Holding Registers* e *Read Input Registers*.

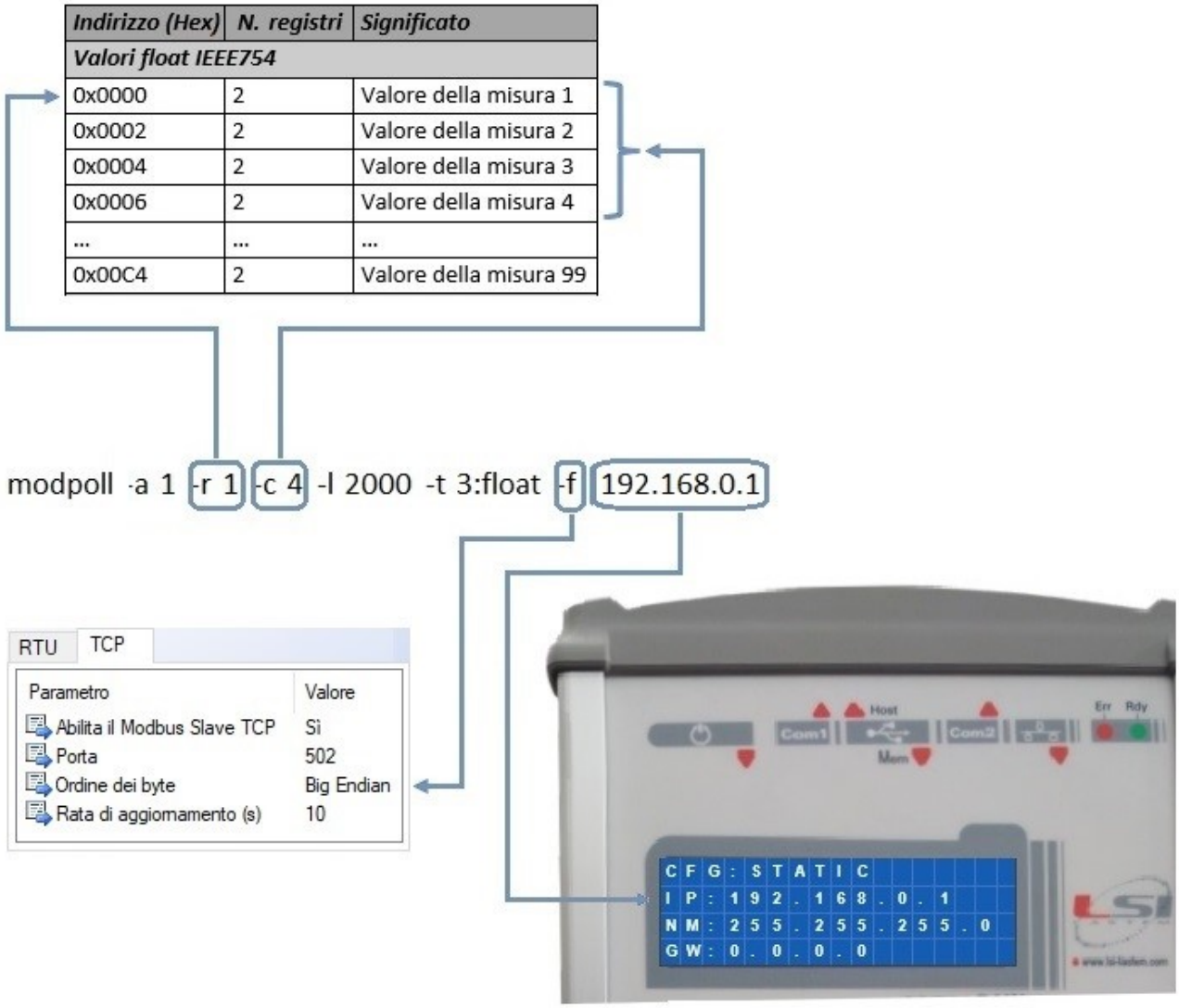
Codice funzione	***Nome funzione***	***Descrizione***
03	Read Holding Registers	Legge gli ultimi valori acquisiti dallo strumento (sia misure analogiche che digitali) e la data ora di sistema
04	Read Input Registers	Come *Read Holding Registers*

Le funzioni sono implementate esattamente come in Modbus RTU. Per maggiori informazioni fare riferimento a §2.3.2 e §2.3.3.

3.4 Verifica con programma modpoll

Come per Modbus RTU, anche i codici funzione 0x03 (*Read Holding Registers*) e 0x04 (*Read Input Registers*) di Modbus TCP possono essere verificati con il programma di simulazione modpoll. I parametri da utilizzare non sono quelli della porta seriale bensì quelli della connessione TCP.

A titolo di esempio si riporta la "costruzione" del comando *Read Input Register* per richiedere i valori delle prime 4 misure di Alpha-Log configurato in modalità Modbus Slave TCP.



E questo è il risultato:

```
modpoll 3.1 - FieldTalk(tm) Modbus(R) Master Simulator
Copyright (c) 2002-2011 proconX Pty Ltd
Visit http://www.modbusdriver.com for Modbus libraries and tools.

Protocol configuration: MODBUS/TCP
Slave configuration...: address = 1, start reference = 1, count = 4
Communication.....: 192.168.0.1, port 502, t/o 1.00 s, poll rate 2000 ms
Data type.....: 32-bit float, input register table
Word swapping.....: Slave configured as big-endian float machine

-- Polling slave... (Ctrl-C to stop)
[1]: 22.599998
[3]: 12.200000
[5]: 1002.700012
[7]: 1002.400024
-- Polling slave... (Ctrl-C to stop)
[1]: 22.599998
[3]: 12.200000
[5]: 1002.700012
[7]: 1002.400024
-- Polling slave... (Ctrl-C to stop)
[1]: 22.599998
[3]: 12.200000…
```

4 Protocollo MQTT

Alpha-Log supporta l'uso del protocollo MQTT per l'invio delle informazioni quali dati, diagnostiche e configurazioni.

MQTT è un protocollo basato su TCP/IP basato sul *publish-subscribe pattern* dove uno o più client si connettono ad un server centrale detto *broker* e possono inviare e ricevere dati inerenti a specifici *topic*. Lo scambio dei dati avviene tramite due principali azioni:

- *publish*: il client comunica al broker di pubblicare i dati inerenti ad un relativo topic;
- *subscribe*: il client chiede al broker di inviagli tutte le informazioni relative ad un topic di interesse.

Ciascun client può inviare informazioni e sottoscriversi a quanti topic desidera senza alcuna limitazione ed il broker si limiterà a fare da "passacarte" inoltrando i messaggi identificati da un certo topic a tutti i client correntemente connessi e sottoscritti a tale topic.

Ad ogni messaggio inviato, oltre a topic e payload, è associato anche un livello di *QoS* e un flag detto *retain*.

Il QoS dice al broker con quale certezza si vuole recapitare il messaggio:

- 0 = At most once
- 1 = At least once
- 2 = Exactly once

Il flag *retain* indica se il messaggio, per il topic assegnato, deve essere mantenuto in memoria dal broker. In tal caso il messaggio verrà distribuito ai client sottoscritti al topic del messaggio non solo in tempo reale ma anche ad una nuova sottoscrizione del client, successiva alla sua pubblicazione. L'ultimo messaggio con retain inviato su ciascun topic andrà a sovrascrivere il messaggio salvato precedentemente con retain sullo stesso topic, mantenendo quindi sempre l'ultimo messaggio arrivato (con retain) per ciascun topic.

Il payload di un messaggio MQTT può avere un qualsiasi formato; Alpha-Log utilizza il formato di testo JSON.

4.1 MQTT su Alpha-Log

AlphaLog è in grado di inviare dati tramite protocollo MQTT ad un broker configurabile.

La configurazione di questo servizio permette la scelta di diversi parametri:

- hostname/ip address del broker [obbligatorio]
- porta, default=1883
- username, default=<vuoto>
- password, default=<vuoto>
- flag per l'invio di diverse tipologie di dati:
  - istantanei
  - elaborati
  - diagnostici
  - allarmi
- in caso venga scelto di inviare dati istantanei, è possibile scegliere una rata di invio (in secondi)

Il topic dei messaggi è composto da una base fissa e non modificabile, seguita da una parte specifica per il tipo di messaggio:

device/<modello>/<seriale>/<azione>

dove:

- *<modello>*: è il codice tecnico del prodotto (ALP001, ALP002, ALP*xxx* a seconda della generazione).
- *<seriale>*: è il seriale di fabbrica del dispositivo o un seriale utente, se specificato nella configurazione.
- *<azione>*: è la parte del topic specifica per ogni tipo di dato (ed esempio, dati istantanei, metrics/inst).

Un topic di esempio, per un dispositivo AlphaLog di seconda generazione che invia dati istantanei e con seriale 21030052 potrebbe essere:

device/ALP002/21030052/metrics/inst.

Tutti i messaggi sono trasmessi con QoS=1 non modificabile dall'utente.

4.2 Organizzazione dei dati

I dati prodotti dallo strumento sono organizzati in:

- dati istantanei: le letture del singolo campionamento della singola variabile (e.g. il valore attuale della temperatura ambiente)
- dati elaborati: l'elaborazione statistica di un dato istantaneo in un intervallo di tempo (e.g. il valore massimo della temperatura ambiente nei 5 minuti dalle 16:00:00 alle 16:04:59)

I dati istantanei non vengono salvati nello strumento e possono essere inviati solamente in tempo reale, senza possibilità di rilettura. I dati elaborati vengono invece mantenuti nella memoria dello strumento e sono scaricabili più volte anche successivamente alla loro creazione.

I dati elaborati prodotti vengono raggruppati per rata di elaborazione: ciascuna rata di elaborazione definisce una base di elaborazione. Le basi di elaborazione sono identificate da un indice *zero-based* deciso automaticamente in fase di configurazione. Ad esempio, avendo tre rate di elaborazione (60”, 300”, 600”) si avranno 3 basi di elaborazione (nell’ordine: 0, 1, 2).

4.2.1 Dati istantanei

Il messaggio relativo ai dati istantanei è un messaggio contenente i valori degli ultimi dati prelevati dal dispositivo, senza alcuna elaborazione statistica ma con le dovute correzioni (linearizzazione, polinomiale, etc.) Il messaggio viene inviato ad un intervallo configurabile e non può essere ritrasmesso.

TOPIC	*device/<modello>/<serial>/metrics/inst*

PAYLOAD	{
	"inst": [
	1022.4,
	14.1,
	0.0,
	287.48
	],
	"time": "2021-10-28T10:04:21"
	}

dove:

- *Inst:* è un array di numeri espressi in virgola mobile contenente i valori istantanei delle misure in quel dato momento, in caso di misura in errore il valore trasmesso è *null*.
- *time*: è il timestamp in UTC del momento di lettura dei dati dal dispositivo.

4.2.2 Dati elaborati

I dati elaborati sono i ricavati da una elaborazione statistica nel tempo (media, minima, massima, deviazione standard, etc.). Questo tipo di messaggio viene inviato ad un intervallo specificato in fase di configurazione, e non coincide necessariamente con la rata di elaborazione.
Per ogni base di elaborazione viene inviato un messaggio contenente tutte le variabili elaborate alla rata di elaborazione associata alla base.

TOPIC	*device/<modello>/<serial>/metrics/elabs*

PAYLOAD	{
	"elab_config_time": "2021-10-27T15:16:33",
	"last_elab_time": "2021-10-28T01:20:00",
	"first_elab_time": "2021-10-28T01:11:00",
	"base": 0,
	"original_filename":"M18050080-C20211027151633-B00-E20211028011100-L20211028012000.txt",
	"serial": "18050080",
	"elab": [
	{
	"items": [
	45.376802,
	9.398528,
	200.0,
	7200.0,
	1014.59,
	1014.62,
	1014.67,
	35.54,
	35.55,
	35.55
	],
	"time": "2021-10-28T01:11:00"
	}
	]
	}

dove:

- *elab_config_time:* è il timestamp della configurazione dei dati.
- *[first|last]_elab_time:* rappresentano i timestamp del primo e dell'ultimo dato del messaggio corrente.
- *base:* è la base di elaborazione alla quale appartengono i dati.
- *original_filename:* è il nome del file di origine del messaggio, così come consultabile sul file system del dispositivo nella cartella dati (*/var/local/ssb/data/history*).
- *serial:* è il seriale di fabbrica del dispositivo
- *elab:* è un array di oggetti json composti da un campo *items* (i valori double risultato della elaborazione statistica).
- *Time:* il timestamp di riferimento per l'elaborazione.

4.2.3 Configurazione dati elaborati

I dati istantanei/elaborati, per essere riconosciuti, hanno bisogno di un messaggio descrittivo che ne permetta la mappatura fra array di double e variabile. Questo tipo di messaggio, inviato con rata di elaborazione specificata in fase di configurazione, contiene informazioni inerenti alle misure trasmesse.

TOPIC	*device/<modello>/<serial>/config/metrics*

PAYLOAD	{
	"MsgUpdateRate": 10,
	"Measures": [
	{
	"MeasKey": "ATM",
	"Name": "ATM",
	"Unit": "",
	"UpdateRate": 5,
	"Prec": 2,
	"ElabSingleGuid": 65535,
	"Elabs": [
	{
	"Rate": 60,
	"Type": "ScalarStat",
	"Elements": "Min, Ave, Max",
	"CoupledGuid": 65535
	}
	],
	"ProbeSerial": "",
	"ProbeCert": "",
	"WMOCode": "",
	"WMOLevel": ""
	},
	{
	"MeasKey": "TEMP",
	"Name": "TEMP",
	"Unit": "",
	"UpdateRate": 10,
	"Prec": 2,
	"ElabSingleGuid": 65535,
	"Elabs": [
	{
	"Rate": 60,
	"Type": "ScalarStat",
	"Elements": "Min, Ave, Max",
	"CoupledGuid": 65535
	},
	{
	"Rate": 300,
	"Type": "ScalarStat",
	"Elements": "Min, Ave, Max",
	"CoupledGuid": 65535
	}
	],
	"ProbeSerial": "",
	"ProbeCert": "",
	"WMOCode": "",
	"WMOLevel": ""
	}
	]
	}

dove:

- *MsgUpdateRate*: è la rata di invio dei dati istantanei
- *Measures*: è un array contenente i dati delle misure configurate (nome, tipologia, id, unità di misura, rate di campionamento) e le elaborazioni configurate per ciascuna di esse. Fra questi parametri i più importanti sono:
    - *Name*: nome della misura (parametro testuale libero)
    - *Unit*: unità di misura (parametro testuale libero)

- *UpdateRate*: rata di campionamento (intero, in secondi, se =0 la misura non effettua campionamenti periodici ma comunica il nuovo valore della misurazione in modo spontaneo)
- *Elabs*: array delle elaborazioni relativa alla misura, raggruppate per tipologia (vettoriale/scalare) e per rata di elaborazione. Ciascun elemento dell'array contiene diversi parametri, fra i più importanti vi sono:
  - *Rate*: rata di elaborazione, in secondi
  - *Type*: tipo di elaborazione (scalare o vettoriale)
  - *Elements*: elaborazioni calcolate su questa misura nell'intervallo di tempo espresso da *Rate* (Inst, Min, Ave, Max, StdDev, Tot, Duration, TimeMin, TimeMax, PrevDir, RisDir, RisVel, StdDevDir, CalmPerc)

Nell'array *Measures* le misure sono nello stesso ordine dei dati istantanei mentre per mappare i dati elaborati c'è bisogno di verificare la sequenza delle informazioni relative alla elaborazione di ogni misura.

4.2.3.1 *Esempio di configurazione*

Nel messaggio *config/metrics* di esempio visto in precedenza è possibile trovare due misure ATM e TEMP che rispecchieranno i soli due valori presenti nell'array delle misure istantanee. Ciascuna misura è campionata rispettivamente ogni 5 e 10 secondi.

Per la misura ATM è stata configurata l'elaborazione al minuto di minima, media e massima dei valori rilevati in quell'intervallo di tempo.

Per la misura TEMP sono state configurate due basi di elaborazione:

- base di elaborazione 0: 60 secondi, con minima, media e massima
- base di elaborazione 1: 300 secondi, con minima, media e massima

Verranno quindi prodotti i seguenti valori elaborati:

- ogni 60 secondi: ATM[min], ATM[avg], ATM[max], TEMP[min], TEMP[avg], TEMP[max]
- ogni 300 secondi: TEMP[min], TEMP[avg], TEMP[max]

Ipotizzando che la rata di invio dati sia ogni 10 minuti (600"), all'invio dei dati verranno inviati i seguenti messaggi di tipo *config/metrics*:

- un messaggio relative alla base di elaborazione 0, dove nell'array *elab* vi saranno presenti 10 elementi, ciascuno con 6 valori all'interno dell'array *items*.
- un messaggio relativo alla base di elaborazione 1, dove nell'array *elab* vi saranno presenti 2 elementi, ciascuno con 3 valori all'interno dell'array *items*.

In caso di mancato invio, i dati verranno messi in una coda e re-inviati successivamente insieme agli eventuali nuovi dati creati.

4.2.4 Dati diagnostici

I dati diagnostici sono i dati ricavati estraendo informazioni da diverse componenti del sistema (OS Linux, data logging e campionamento, comunicazioni, orologi, etc.)

TOPIC	*device/<modello>/<serial>/status/diagnostic*

PAYLOAD


```
{
 "network": {
  "interfaces": {
   "eth0": {
    "tx_bytes": "241758",
    "ip": "192.168.1.69",
    "mac": "00:d0:69:4d:6f:de",
    "rx_bytes": "505146",
    "internet": {
     "status": true,
     "ip": "80.180.46.234"
    }
   }
  }
 },
 "device_space": {
  "used": 2461060.0,
  "perc_used": 81.2,
  "free": 569740.0,
  "is_full": false,
  "perc_free": 18.8,
  "total": 3030800.0
 },
 "com_stats": {
  "up_time": 84150.0,
  "COM": [
   {
    "tx_frames": 0,
    "name": "COM1",
    "rx_bytes": 0,
    "rx_frames": 0,
    "rx_failed": 0,
    "tx_bytes": 0
   },
   {
    "tx_frames": 0,
    "name": "COM2",
    "rx_bytes": 0,
    "rx_frames": 0,
    "rx_failed": 0,
    "tx_bytes": 0
   },
   {
    "tx_frames": 0,
    "name": "COM3",
    "rx_bytes": 0,
    "rx_frames": 0,
    "rx_failed": 0,
    "tx_bytes": 0
   },
   {
    "tx_frames": 0,
    "name": "COM4",
    "rx_bytes": 0,
    "rx_frames": 0,
    "rx_failed": 0,
    "tx_bytes": 0
   },
```

(continua) PAYLOAD	{
	"tx_frames": 0,
	"name": "COM5",
	"rx_bytes": 0,
	"rx_frames": 0,
	"rx_failed": 0,
	"tx_bytes": 0
	},
	{
	"tx_frames": 11425,
	"name": "COM6",
	"rx_bytes": 133377,
	"rx_frames": 11426,
	"rx_failed": 0,
	"tx_bytes": 230795
	}
	]
	},
	"time": {
	"sbc": "2021-10-28T12:40:11",
	"sbc_uptime": "14:40:11 up 43 min, 1 user, load average: 0.25, 0.25, 0.32",
	"ssb": "2021-10-28T14:40:12"
	}
	}

dove:

- *network.interfaces*: è un dizionario di interfacce di rete, dove la chiave è il nome di sistema dell'interfaccia e il valore sono le sue proprietà, con i seguenti campi:
  - *tx_bytes, rx_bytes:* byte trasmessi e ricevuti.
  - *ip:* indirizzo IP locale.
  - *mac:* MAC address.
  - *internet.status* e *internet.ip:* [opzionali] indicano se dalla periferica è raggiungibile internet e, in caso, con quale IP.
- *device_space*: contiene informazioni sullo spazio disponibile nel dispositivo, dove:
  - *is_full*: permette di capire non solo se il disco è pieno per i troppi dati ma anche se vi sono troppi file in coda di invio
  - *used, free*: numero di byte totali, in uso e liberi
  - *perc_used, perc_free*: percentuale dello spazio in uso e libero
- *com_stats*: contiene informazioni prese dal sistema di campionamento di basso livello: *uptime* è il numero di secondi a partire dall'accensione del sistema mentre *COM* è l'array contenente le statistiche di comunicazione delle varie porte COM disponibili. Ciascun oggetto relativo alla porta COM è composta dalle seguenti proprietà:
  - *name*: nome della porta COM (COM1, COM2, etc.)
  - *tx_frames, rx_frames, rx_failed*: numero di frame inviati, ricevuti, in errore
  - *tx_bytes, rx_bytes*: numero di bytes totali trasmessi e ricevuti
- *time*: contiene informazioni sugli orologi del sistema:
  - *sbc*: è il timestamp in UTC della scheda SBC
  - *ssb*: è il timestamp con timezone della scheda SSB
  - sbc_uptime: contiene informazioni sull'avvio del sistema