Protocollo di comunicazione del sistema di acquisizione dati Kinemetrics/Quanterra Q330

Sergio Guardato
Giovanni Iannaccone

Istituto Nazionale di Geofisica e Vulcanologia
Sezione di Napoli "Osservatorio Vesuviano"

Open File Report n. 2 - 2007

INDICE

Indice delle Figure e delle Tabelle

Riassunto

Nel presente rapporto è descritto il protocollo di comunicazione QDP (Quanterra Data Protocol) implementato dalla Quanterra-Kinemetrics sulla linea di acquisitori Q330 e non direttamente deducibile dalla documentazione rilasciata dalla casa madre.
In particolare è descritta dettagliatamente la procedura di comunicazione per lo scambio dati tra Q330 e qualunque data-client, con l'elenco completo di tutti i possibili comandi diretti al Q330 ed i comportamenti in risposta ad essi associati.
Il presente rapporto si propone, quindi, di fornire una documentazione sulla struttura del protocollo di comunicazione a supporto con i manuali a corredo della strumentazione di questa linea di prodotti.

Descrizione generale del Q330

Il Quanterra Q330, della Kinemetrics Inc., è un sistema di acquisizione dati ad alta risoluzione ed a basso consumo (< 0.8W – incluso GPS). E' ampiamente utilizzato sia per acquisizione in locale che in telemetria, in quest'ultimo caso con l'unità esterna di registrazione su disco PB14F-Baler.

Le caratteristiche generali del Q330 sono:

- 3 o 6 canali di acquisizione (tensione massima di ingresso 40V_{pp} con guadagno unitario);
- amplificatore con guadagno 1÷30 (selezionabile per ciascun canale);
- ADC a 24 bit di risoluzione del tipo delta-sigma;
- Digital Signal Processor integrato;
- 8Mb di memoria RAM;
- ricevitore GPS integrato;
- sensore di temperatura integrato e monitoraggio della tensione di alimentazione (12Vdc);
- gestione ottimizzata dell'energia;
- controllo e calibrazione dei sensori (mass re-centering, etc);
- tre interfacce seriali;
- interfaccia IrDA;
- interfaccia Ethernet 10Base-T.

Nella figura seguente è mostrato il Q330 con l'indicazione dei nomi dei connettori a pannello su di esso alloggiati ed il codice corrispondente.

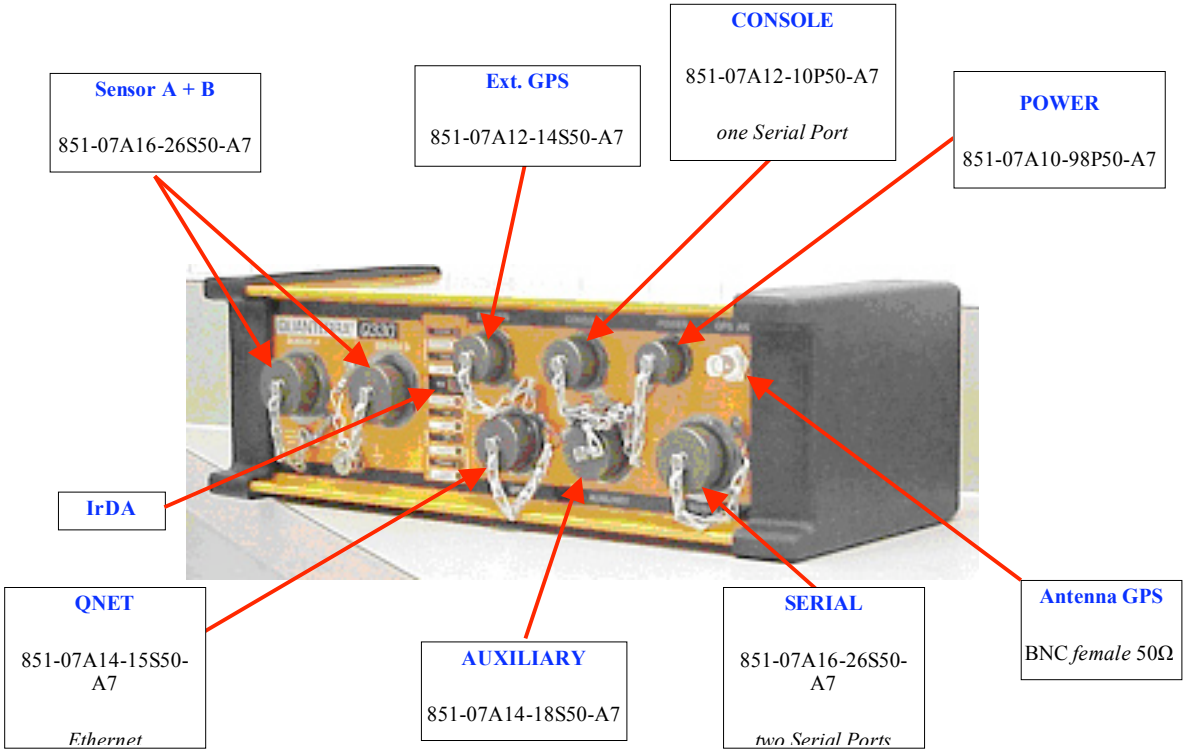

Figura 1: connettori e porte del Q330.

Il Q330 possiede un'interfaccia Ethernet 10Base-T e tre porte seriali (RS-232C single-ended, full-modem control).
La prima è fisicamente disponibile sul connettore denominato QNET – come visibile in figura – mentre una delle tre porte seriali è fisicamente disponibile sul connettore null-modem denominato CONSOLE (RS-232C, UDP framed SLIP, no CNP protocol – a tal proposito si veda il documento denominato "Connector Description for Q330.pdf – Rev. 18B").
Le altre due porte seriali, dello stesso tipo presente sul connettore CONSOLE, sono fisicamente disponibili sul connettore denominato SERIAL.

In realtà esistono altre tre porte seriali: una di queste è appositamente dedicata alle funzionalità del sistema GPS ed è fisicamente disponibile sul connettore denominato Ext. GPS, assieme ad una porta seriale dedicata ancora al GPS di tipo RS-422; l'altra, fisicamente disponibile sul connettore denominato POWER, è solo una porta seriale di replica, che si attiva se la porta CONSOLE o l'interfaccia ad infrarossi (IrDA) è attiva.

I connettori SENSOR A e SENSOR B sono per i sensori sismici, generalmente accelerometro e sensore larga-banda.

La comunicazione con il Q330 può avvenire attraverso due delle tre porte seriali fisicamente disponibili o tramite Ethernet. Ogni interfaccia, seriale o Ethernet, possiede un proprio indirizzo IP, ma questi possono anche essere tra loro uguali.

Le tre porte seriali fisicamente disponibili utilizzano tutti e nove i pins (se il connettore che viene usato per la comunicazione è del tipo DB9), poiché deve essere prevista la possibilità di comunicare con il controllo del flusso hardware e software (RTS/CTS, DTR/DSR/DCD, RI). In particolare, il collegamento del Data Processor con il Q330 avviene fisicamente tramite cavo provvisto di connettore, tipo militare (851-06EC12-10S50), collegato alla porta CONSOLE del Q330.

Nei paragrafi successivi con il termine Data Processor, DP, sarà indicato il generico dispositivo client di connessione con il Q330

Il protocollo di comunicazione

Il Q330 possiede quattro porte mappate in memoria (logiche, non fisiche) dedicate per misure in telemetria, chiamate **Data Port**. La comunicazione su ognuna di queste porte, dotate di opportuni buffers (sino ad 8MB condivisi), avviene tramite un'implementazione di protocollo UDP/IP su porta seriale con trasferimento di pacchetti QDP (***Q**uanterra **D**ata **P**acket*). Si tratta dello SLIP, un protocollo che ricostruisce il frame UDP/IP su porta seriale. Il metodo usato dal protocollo è come quello denominato DSS (Data Subscription Server). Inoltre, il Q330 è abilitato a rispondere al comando *ping*; a tale scopo è stato implementato anche l'echo ICMP.

Un pacchetto QDP consiste di header's e di dati, e può essere di lunghezza variabile.

Ogni Data Port è provvista di un registro MTU (Maximum Transmission Unit) programmabile; esso determina il numero massimo di bytes trasferibili con un pacchetto. Tale lunghezza presenta un limite superiore di 576 bytes del pacchetto intero, vale a dire di massimo 536 bytes di dati per pacchetto (20+8+12+ 536B per i dati) ed inferiore, minimo, di 276 bytes (20+8+12+ 236B per i dati).

Il 'datagramma' di un pacchetto QDP consiste in quattro campi di seguito denominati con le rispettive lunghezze:

Campo	*lunghezza (bytes)*
IP header	20
UDP header	8
QDP header	12
DATA	236 ÷ 536

Tabella 1: pacchetto QDP

In dettaglio visivo:

Figura 2: QDP frame.

Il campo QDP deve essere semplicemente interpretato come un header per il campo DATA.

I campi IP e UDP sono quelli 'classici' dell'incapsulamento dei dati derivanti dalla comunicazione con protocollo UDP/IP.

Il campo QDP (costituito da 12 bytes) è a sua volta suddiviso in sottocampi di dimensione diversa; essi sono:

QDP header

CRC	Cmd	Ver	Len	SeqNum	ACKNum
4	1	1	2	2	2

Figura 3: QDP header frame.

- **CRC**
  è un valore di CRC, da calcolare sull'*intero* pacchetto QDP secondo le indicazioni fornite dalla Kinemetrics, costituito da 4 bytes (tipo longint signed);

- Command (**Cmd**)
  è il codice esadecimale del comando inviato dal DP (o della risposta ad opera del Q330) costituito da 1 byte ($00 ÷ $FF) – come da tabella che segue (tipo char);

- Version (**Ver**)
  questo numero, costituito da 1 byte, viene usato per consentire delle revisioni sul protocollo (attualmente vale 2, ovvero $**02** – è fissato a tale valore sino a che la versione del software è 1.xx) - (tipo char);

- Length (**Len**)
  è costituito da 2 bytes, e rappresenta il numero massimo di bytes di DATA da trasferire, compreso tra 0 e 536 ($0000 ÷ $0218), senza includere i 40 bytes di header's (=20+8+12) – (tipo word).

  Questo valore contenuto in tale campo è dato dal numero contenuto nel campo *LenUDP* dell'header UDP sottratti gli 8 bytes della lunghezza del campo UDP e sottratti i 12 bytes della lunghezza del campo *QDP* (Len = LenUDP – 20 bytes)

- Sequence Number (**SeqNum**)
  è un numero, costituito da 2 bytes ($0000 ÷ $FFFF), da incrementare di uno di volta in volta, per ogni pacchetto, indicante al Q330 il numero (label) del pacchetto attuale spedito (tipo word);

- Acknowledge Number (**ACQNum**)
  è un numero, costituito da 2 bytes ($0000 ÷ $FFFF), per fare l'acknowledgement dell'ultimo datagramma QDP ricevuto (tipo word).

  Se si tratta di un pacchetto di comando, l'ACQNum è il numero del datagramma appena ricevuto; invece, se si tratta di un pacchetto di dati, allora questo è il numero dell'ultimo datagramma ricevuto.

I comandi

I comandi sono suddivisi in base al modulo che li trasmette, DP o acquisitore Q330. In ogni caso essi sono raggruppati in quattro categorie denominate, rispettivamente: **Basic**, **Memory**, **Advanced** e **Tertiary**.
In particolare, i comandi che il DP invia all'acquisitore tramite data port, sono in quantità e per tipologia di sotto elencati in tabella:

Command type	*Qty number*
Basic	39
Memory	5
Advanced	14
Tertiary	2

Tabella 2: comandi.

Oltre questi figurano un comando di apertura Data Port, ed il segnale di Data Acknowledge.

Di seguito viene mostrata la tabella completa dei comandi che il DP invia al Q330, e le possibili risposte di quest'ultimo, distinti per tipologia, con l'indicazione del codice esadecimale per la codifica (un byte), il nome, la descrizione breve, il numero dei parametri necessari ed eventuali note descrittive.

Per la descrizione dettagliata dei singoli comandi e\o delle possibili risposte si rimanda al documento della Kinemetrics Q330 Communications Protocol Rev. 15 – System Software Ver. 1.68 – 23-Feb-2005.

DP to Q330					
code	***name***	***description***	***parameters***	***responses***	***notes***
			…		
0A	DT_DACK	Data Acknowledge	more		
0B	DT_OPEN	Open Data Port	none		
			Basic		
10	C1_RQSRV	Request Server Registration	Serial Number (2 * 4 bytes)		Broadcast ok
11	C1_SRVRSP	Server Response	more		Broadcast ok
12	C1_DSRV	Delete Server	Serial Number (2 * 4 bytes)	C1_CACK ($A0)	Broadcast ok
13	C1_SAUTH	Set Authorization Codes	more		Broadcast ok
14	C1_POLLSN	Poll for Serial Number	more		
15	C1_SPHY	Set Physical Interfaces	more		Broadcast ok
16	C1_RQPHY	Request Physical Interfaces	none	C1_PHY ($A4)	Broadcast ok
17	C1_SLOG	Set Data Port	more		
18	C1_RQLOG	Request Data Port	Data Port Number (2 bytes)	C1_LOG ($A5)	
19	C1_CTRL	Control Q330 Operation	Flags (2 bytes)		Broadcast ok
1A	C1_SGLOB	Set Global Programming	more		
1B	C1_RQGLOB	Request global Programming	none	C1_GLOB ($A6)	
1C	C1_RQFIX	Request Fixed Values after Reboot	none	C1_FIX ($A7)	
1D	C1_SMAN	Set Manufacturer's Area	more		
1E	C1_RQMAN	Request Manufacturer's Area	none	C1_MAN ($A8)	
1F	C1_RQSTAT	Request Status	Request Bitmap (4 bytes)	C1_STAT ($A9)	
20	C1_WSTAT	Write to Status Port	Spare (2 bytes) + New Value (2 bytes)		
21	C1_VCO	Set VCO	New Current VCO Value (2 bytes) + PLL Flag (2 bytes)		Configuration Port only
22	C1_PULSE	Pulse sensor control line(s)	Sensor Control Bitmap (2 bytes) + Duration (10ms int) (2 bytes)		
23	C1_QCAL	Start QCAL330 Calibration	more		
24	C1_STOP	Stop Calibration	none		
25	C1_RQRT	Request Routing Table	none	C1_RT ($AA)	
26	C1_MRT	Modify Routing Table	IP Address (2 bytes) + New Physical Interface (2 bytes) + New Data Port (2 bytes)		
27	C1_RQTHN	Request Thread Names	none	C1_THN ($AB)	
28	C1_RQGID	Request GPS ID Strings	none	C1_GID ($AC)	
29	C1_SCNP	Send CNP Message	more	C1_RCNP ($AD)	
2A	C1_SRTC	Set Real Time Clock	more		
2B	C1_SOUT	Set Outputs Bits	New Bitmap (2 bytes)		
2C	C1_SSPP	Set Slave Processor Parameters	more		
2D	C1_RQSPP	Request Slave Processor Parameters	none	C1_SPP ($AE)	
2E	C1_SSC	Set Sensor Control Mapping	(8 x 4 bytes)		
2F	C1_RQSC	Request Sensor Control Mapping	none	C1_SC ($AF)	

…continua

...segue

30	C1_UMSG	Send User Message	(2 x 4 bytes)		
33	C1_WEB	Set Web Server Link	(2 x 4 bytes)		
34	C1_RQFLGS	Request Combination Packet	Data Port Number (2 bytes)	C1_FLGS ($B1)	
35	C1_RQDCP	Request Digitizer Calibration Packet	none	C1_DCP ($B2)	
36	C1_RQDEV	Request CNP Device Info	none	C1_DEV ($B3)	
37	C1_SDEV	Set Device Options	(2 x 4 bytes)		
38	C1_PING	Ping Q330	more		DP ←-> Q330
Memory					
40	C1_SMEM	Set Memory Contents	more		
41	C1_RQMEM	Request Memory Contents	more	C1_MEM ($B8)	
42	C1_ERASE	Erase Flash Sectors	(2 x 2 bytes)		
43	C1_RQMOD	Request Memory Module Map	none	C1_MOD ($B9)	
44	C1_RQFREE	Request Free Flash Memory	none	C1_FREE ($BA)	
Advanced					
50	C2_SPHY	Set Physical Interface	more		
51	C2_RQPHY	Request Physical Interface	Physical Inteface Number (2 bytes)	C2_PHY ($C0)	
52	C2_SGPS	Set GPS Parameters	more		
53	C2_RQGPS	Request GPS Parameters	none	C2_GPS ($C1)	
54	C2_SWIN	Set Recording Windows	more		
55	C2_RQWIN	Request Recording Windows	none	C2_WIN ($C2)	
57	C2_SAMASS	Automatic Mass Re-centering	more		
58	C2_RQAMASS	Request Automatic Mass Re-centering	none	C2_AMASS ($C4)	
59	C2_SBPWR	Set Bale Power and Dialer Control	Physical Interface (2 bytes) + Flag and Timeout (2 bytes)		
5A	C2_BRDY	Baler Ready	more		* Data Vacuum
5D	C2_REGCHK	Registration Check	IP Address (4 bytes)	C2_REGRESP ($C9)	
5F	C2_RQQV	QuickView Request	more	C2_QV ($CA)	
60	C2_RQMD5	Request MD5 Result		C2_MD5 ($CB)	
69	C2_TERC	Tertiary Commands	more	C2_TERR ($6A)	
Tertiary					
(1)	C3_RQANNC	Request Announce Structure	more	C3_ANNC (0)	
(2)	C3_SANNC	Set Announce Structure		C1_CACK ($A0)	

Tabella 3: comandi del DP.

I comandi di tipo Memory consentono l'accesso alla memoria del Q330: data, program, etc. E' possibile trasferire sino a 484 bytes di dati dalla memoria del Q330.

Le possibili risposte del Q330

Q330 to DP			
code ($)	***name***	***description***	***infos***
00	DT_DATA	Data Record	
06	DT_FILL	Fill Packet	Fill Record Sequence Number (4 bytes)
Basic			
A0	C1_CACK	Command Acknowledge	none
A1	C1_SRVCH	Server Challenge	more
A2	C1_CERR	Command Error	Error Code (2 bytes)
A3	C1_MYSN	My Serial Number	more
A4	C1_PHY	Physical Interfaces	
A5	C1_LOG	Data Port	
A6	C1_GLOB	Global Programming	
A7	C1_FIX	Fixed Value after Reboot	more
A8	C1_MAN	Manufacturer's Area	
A9	C1_STAT	Status	more
AA	C1_RT	Routing Table	more
AB	C1_THN	Thread Names	
AC	C1_GID	GPS ID Strings	(9 x 4 bytes)
AD	C1_RCNP	CNP Reply Message	(2 x 4 bytes)
AE	C1_SPP	Slave Processor Parameters	
AF	C1_SC	Sensor Control Mapping	
B1	C1_FLGS	Combination Packet	more
B2	C1_DCP	Digitizer Calibration Packet	(6 x 2 x 4 bytes)
B3	C1_DEV	CNP Device Info	more
Memory			
B8	C1_MEM	Memory Contents	more
B9	C1_MOD	Memory Module Map	more
BA	C1_FREE	Free Flash Memory	(2 x 4 bytes)
Advanced			
C0	C2_PHY	Physical Interfaces	
C1	C2_GPS	GPS Parameters	
C2	C2_WIN	Recording Windows	
C4	C2_AMASS	Automatic Mass Re-centering	
C6	C2_BACK	Baler Acknowledge	
C9	C2_REGRESP	Registration Response	(4 bytes)
CA	C2_QV	QuickView Response	more
CB	C2_MD5	MD5 Result	128 bits
6A	C2_TERR	Tertiary Responses	more
Tertiary			
00	C3_ANNC	Announce Structure	more

Tabella 4: le risposte del Q330.

. . . . . .

others					
code ($)	*name*	*description*	*parameters*	*responses*	*notes*
C5	C2_POC	Point of Contact	none	none	Q330 -> POC
C7	C2_VACK	Vacuum Acknowledge	none	C2_BRDY ($5A)	VACC -> DP
5B	C2_BOFF	Baler Off	Power-off Delay in one second increments (2 bytes)	C1_CACK ($A0)	Baler -> Q330
C8	C2_BCMD	Baler Command	more	C2_BRESP ($5C)	VACC -> DP
5C	C2_BRESP	Baler Response	more	none	DP -> VACC
5E	C2_INST	Installer Command	more	C2_POC ($C5)	Installer -> Q330 DB9 Interface

Tabella 5: altri comandi.

Nell'utilizzo del Q330 occorre prevedere che il DP, nel trasmettere i comandi verso il Q330, possa non ricevere risposta da quest'ultimo; pertanto bisogna creare un meccanismo tale che, se non ci dovesse essere nessuna risposta ad un comando inviato dal DP al Q330 (laddove una risposta è prevista) in un certo tempo (timeout), allora il comando viene rispedito. Se anche in quest'ultimo caso non ci dovesse essere risposta, allora la connessione viene chiusa (la porta seriale chiusa) etc.

Se trascorre un certo tempo di inattività nella comunicazione (programmabile), allora il Q330 si disconnette automaticamente. Il DP per poter comunicare nuovamente con il Q330 deve inizializzare la procedura di comunicazione come descritto nel successivo paragrafo.

Si noti che la comunicazione tra DP e Q330 prevede talvolta un handshake e, come anzidetto, che – in alcuni casi - per ogni comando inviato dal DP al Q330 bisogna incrementare il Sequence Number (**SeqNum**), in modo tale che alla risposta da parte del Q330 al DP, quest'ultimo possa fare l'acknowledgement del comando (pacchetto) precedentemente trasmesso, con le modalità già viste quando si è parlato dello ACQNum per l'header QDP.

Procedura di comunicazione

Premessa

Anche il DP va impostato con un suo indirizzo IP ed un valido indirizzo per il Data Port. Questi parametri di configurazione interni possono essere riconfigurabili.

Nella comunicazione bisogna tenere conto della procedura di registrazione iniziale che rispetti questa sequenza:

- Registrazione del DP sul Q330;
- Richiesta, da parte del DP, delle informazioni e dei pacchetti di configurazione del Q330;
- Richiesta, da parte del DP, degli stati iniziali del Q330;
- Richiesta, da parte del DP, di apertura della porta UDP del Q330;
- Richiesta periodica, da parte del DP, dello stato del Q330;
- Elaborazione, da parte del DP, dei dati e dei "blockettes" trasmessi dal Q330.

Nel seguito è descritta in dettaglio la sequenza.

Registrazione

La registrazione del DP sul Q330 può essere fatta utilizzando la porta di configurazione UDP o una delle logical data ports. Il DP, per poter comunicare, deve conoscere del Q330:

- IP address;
- Base Address della Logical Port Number (solitamente è 5000d);
- Serial Number;
- Authentication Code.

Quest'ultimo viene inviato dal Q330 usando la codifica MD5. Essa consiste nel generare una lunga stringa di caratteri, usando il campo DATA dei pacchetti, in congiunzione con l'Authorization Code per poi 'passare' questa all'algoritmo MD5 per generare un risultato a 128 bit. Sia il DP che il Q330 eseguono lo stesso calcolo e, se i risultati corrispondono, si presume che il DP conosce l'Authentication Code del Q330, e pertanto potrà comunicare con il Q330.

Esempio: (i numeri esadecimali sono preceduti da un segno di dollaro $)

Per il Q330:

- Serial Number: $010054A3498255F2 – 8 bytes;
- Authentication Code: $A7340ACB2490ED64 – 8 bytes;
- IP Address: 216.120.82.34 ($D8785222);
- si supponga di voler utilizzare la logical data port all'UDP port number 5002d: $138A.

per il DP:

- IP Address: 123.234.210.024 ($7BEAD218);
- UDP port number 1344d per il controllo della connessione: $0540.

Ed ecco come procede la registrazione:

1. Il DP invia al Q330 un pacchetto QDP contenente: l'IP e l'UDP port number del DP ($7BEAD218:$0540), l'IP e l'UDP port number del Q330 ($D8785222:$138A), il comando ($10) di richiesta registrazione C1_RQSRV, e poi il numero seriale del Q330 ($010054A3498255F2);

2. il Q330 risponde con un altro pacchetto QDP contenente: la risposta C1_SRVCH ($A1) al comando, con un valore numerico casuale (ad esempio: $1234567890ABCDEF – 8 bytes), l'IP e l'UDP port number del DP ($7BEAD218:$0540), ed infine, il numero di registrazione del DP (ad esempio: $0012 – 2 bytes); eventualmente, se si usa la porta Ethernet, il MAC address del DP stesso;

3. il DP genera un numero casuale, di 8 bytes, da usare nei prossimi calcoli (sia esso, ad esempio: $FEDCBA0987654321);

4. il DP assembla una lunga stringa in base alle regole della risposta C1_SRVRSP:

   '1234567890ABCDEF7BEAD21805400012A7340ACB2490ED64010054A3498255F2FEDCBA0987654321'

   e su questa stringa va ad eseguire l'algoritmo MD5, ottenendo un risultato lungo 16 bytes;

5. a questo punto il DP invia il pacchetto QDP con il messaggio C1_SRVRSP ($11) seguito da:

   - il Serial Number del Q330 ($010054A3498255F2);
   - il numero casuale generato ad opera del Q330 ($1234567890ABCDEF);
   - l'IP del DP ($7BEAD218);
   - l'UDP port number del DP ($0540);
   - il numero di registrazione del DP ($0012);
   - il numero casuale generato ad opera del DP ($FEDCBA0987654321)
   - e, finalmente, il risultato dell'algoritmo MD5 (128 bit – 16 bytes).

6. il Q330 preleva queste informazioni ed effettua lo stesso calcolo con l'algoritmo MD5 e confronta il suo risultato con quello inviato dal DP.
   Se il confronto trova risultati diversi, il Q330 invia una risposta di errore (Invalid Registration Response – C1_CERR ($A2)).
   Invece, se il confronto trova gli stessi risultati, allora il Q330 invia l'acknowledge C1_CACK ($A0) in risposta. A questo punto l'IP address, il numero di porta UDP ed, eventualmente, il MAC address del DP vengono registrati (normalmente) per quel Data Port!

   Per motivi di sicurezza, il Q330 impiega circa un secondo per fare l'acknowledge.

7. Il DP è pronto per inviare al Q330 varie strutture e comandi dati in relazione a ciò che si vuole fare.

Nei paragrafi successivi vengono riportati degli esempi di codice, in C, sulle strutture dei dati da usare per il processo di registrazione.

Richiesta informazioni e parametri di configurazione

Dopo che la procedura di registrazione è andata a buon fine, il DP invia al Q330 il comando C1_RQFLGS ($34); il Q330 risponde con il messaggio C1_FLGS ($B1) contenente varie informazioni che il DP deve memorizzare da qualche parte per poi poterle utilizzare successivamente.

Richiesta degli stati iniziali

Dipende dalla specifica richiesta dello stato da visionare.

Apertura della porta UDP

Il DP invia il pacchetto QDP con il comando DT_OPEN ($0B), dal data port del DP al data port del Q330. Con questo comando, il DP chiede al Q330 qual'è l'indirizzo (il port number) della porta UDP del Q330 da cui sono inviati i dati, che il Q330 abilita anche per l'invio degli stessi.
Se non si ottiene nessun pacchetto di dati in risposta dopo circa 100 secondi, è necessario che il DP rispedisca nuovamente questo comando.
Ovviamente il comando è accettato dal Q330 solo se proviene dallo stesso indirizzo IP con cui il DP è stato registrato.

Richiesta periodica dello stato

Dipende dalla specifica richiesta dello stato da tenere sott'occhio e dalla frequenza con cui si decide di farlo.

Elaborazione dei dati e dei "blockettes" ricevuti

I pacchetti UDP contenenti dati, inviati dal Q330 al DP, siano essi DT_DATA ($00) o DT_FILL ($06), arrivano sulla Data Port del DP. Il DP elabora le informazioni in essi contenute ed invia, per ogni pacchetto dati ricevuto, il pacchetto DT_DACK ($0A).
Ogni pacchetto ricevuto dal DP, Data o Fill, inizia con un record di 4 bytes contenente un Sequence Number. Data e Fill posseggono Sequence Number tra loro diversi; ad ogni modo, il Fill Record Sequence Number può essere ignorato. Ogni acquisizione della durata di un secondo del Q330 può essere contenuta in uno o più pacchetti, ognuno dei quali possiede lo stesso Data Record Sequence Number.

Il pacchetto ricevuto è suddiviso in 'blockette' descritti nel documento precedentemente citato a pagina 9.

Cenni sul protocollo UDP/IP

Dal momemto che il protocollo di comunicazione QDP del Q300 è un protocollo che è incapsulato nel protocollo UDP/IP (***U**ser **D**atagram **P**rotocol* - ***I**nternet **P**rotocol*), è opportuno descrivere brevemente la sua struttura.

Come noto si tratta di un protocollo non orientato alla connessione, ovvero senza handshake e senza controllo e gestione degli errori, senza neppure controllo del flusso o riordino dei pacchetti; quindi più veloce del TCP/IP ma meno affidabile. Per certe applicazioni, come la nostra, può andar anche bene.
La trasmissione di un pacchetto UDP lungo una socket avviene incapsulandolo all'interno di un pacchetto IP. Giunto a destinazione, il pacchetto viene inviato alla porta di destinazione indicata nell'intestazione (header) UDP.

L'incapsulamento dei dati nel protocollo UDP/IP su rete Ethernet è del tipo seguente:

Figura 4: UDP/IP frame.

dove ogni campo assume il significato e la lunghezza in bytes qui di seguito specificate:

Campo	*lunghezza (bytes)*
Ethernet header	**14**
IP header	**20**
UDP header	**8**
tftp header	4
data	n
Ethernet footer	4

Tabella 6: campi UDP/IP.

Nel protocollo QDP della Quanterra il campo finale dell'UDP/IP (*Ethernet footer*), ed il campo *tftp header* sono mancanti.

Il campo **IP header** è così suddiviso:

IP header

LIP	ToS	LenIP	Id	Fr	TTL	type	chk	IPs	IPd
1	1	2	2	2	1	1	2	4	4

Figura 5: IP header frame.

- **LIP**
  Indica sia la lunghezza dell'intestazione che la versione del pacchetto IP.

  Il valore di lunghezza deve essere inserito diviso per quattro (il numero viene salvato attraverso i 4 bit meno significativi). Siccome la lunghezza di un pacchetto IP normalmente è di 20 bytes, allora in questo campo solitamente va inserito il valore 5.

  Sino ad oggi esistono due versioni IP: la *IPv4* (4 byte per l'indirizzo IP) e la *Ipv6* (16 byte per l'indirizzo IP); la versione più diffusa è la *IPv4* anche se si sta iniziando ad utilizzare la nuova versione.
  Quindi questo campo, di un byte, va riempito con il valore $**45**.

- Type Of Service (**ToS**)
  Fornisce diverse caratterestiche al datagramma IP che verranno poi utilizzate dai router per l'instradamento dei pacchetti (affidabilità, basso ritardo...). Solitamente vale $**00**.

- **Len**
  Lunghezza totale dell'intero datagramma (pacchetto IP + pacchetto UDP + pacchetto QDP + DATA) - di due bytes.

- **Id**
  Numero identificativo del pacchetto IP (due bytes).

- **Fr**
  Questo campo, lungo due bytes, è utilizzato quando il pacchetto viene frammentato dai router per poter essere trasportato lungo la rete.

- Time To Live (**TTL**)
  Rappresenta il massimo numero di router su cui può passare il pacchetto prima di essere considerato perso. Ogni router su cui passa il pacchetto decrementa questo valore e quando arriva a zero il pacchetto viene scartato.

- **type**
  Indica il tipo di pacchetto che è incapsulato all'interno del pacchetto IP.

  Può valere 6 nel caso di un pacchetto TCP, 17 per UDP, 1 per ICMP oppure può assumere il numero di un qualsiasi altro protocollo. (un byte = $**11** = 17d)

- **chk**
  E' una checksum che permette ai router di rilevare eventuali errori nel pacchetto.

- **IPs**
  indirizzo IP sorgente (DP o Q330) – quattro bytes.

- **IPd**
  indirizzo IP destinatario (Q330 o DP) – quattro bytes.

Il campo **UDP header** è così suddiviso:

UDP header

SP	Dest. P.	LenUDP	ChkSum
2	2	2	2

Figura 6: UDP header frame.

- Source Port (**SP**)
  campo di 2 byte contenente il numero della porta UDP di origine dell'host sorgente (il DP o il Q330);

- Destination Port (**Dest. P.**)
  campo di 2 byte, contenente il numero della porta UDP di destinazione del pacchetto sull'host remoto (il Q330 o il DP);

- Lenght (**LenUDP**)
  campo di 2 byte, contenente la lunghezza in byte dell'intestazione UDP (8 bytes) sommata a quella dell'intestazione QDP (12 bytes) più quella dei dati;

- Checksum (**ChkSum**)
  campo di 2 byte, utilizzato per verificare l'integrità dei dati trasportati.

  Questa checksum è generalmente diversa dal valore di checksum contenuto nell'header IP.

Le porte logiche UDP disponibili nel Q330 sono quattro ed i buffer per i dati sono separati da quelli per i controlli, come riportato nella tabella seguente:

UDP Data Port	
address	name
b.a.	Configuration
b.a. + 1	Special Functions
b.a. + 2	Control Port 1
b.a. + 3	Data Port 1
b.a. + 4	Control Port 2
b.a. + 5	Data Port 2
b.a. + 6	Control Port 3
b.a. + 7	Data Port 3
b.a. + 8	Control Port 4
b.a. + 9	Data Port 4

(base address b.a. = 5330d)

Tabella 7: UDP Data Ports.

Calcolo del CRC

Il CRC viene calcolato e generato mediante l'ausilio di una look-up-table per la velocità di comunicazione tra DP e Q330 mediante l'ausilio di un algoritmo sviluppato dalla Kinemetrics Inc.
Di seguito viene mostrata una procedura di calcolo scritta in linguaggio C, come riportato nel manuale del Q330:

```
#define CRC_POLYNOMIAL  1443300200L

typedef unsigned long tcrctab[256];
tcrctab crctable ;
typedef union lbit {
  longint l;
  struct {
    byte b0, b1, b2, b3;
  } U1;
} lbit;
```

```
Static Void gcrcinit()
{
  int count, bits;
  unsigned long tdata, accum;

  for (count = 0; count <= 255; count++) {
    tdata = count << 24;
    accum = 0;
    for (bits = 1; bits <= 8; bits++) {
      if ((tdata ^ accum) < 0) {
        accum = (accum << 1) ^ CRC_POLYNOMIAL;
      else
        accum <<= 1;
      }
      tdata <<= 1;
    }
    accum = lswap(accum);
    crctable[count] = accum;
  }
}


Static unsigned int gcrccalc(b, len)
char *b;
unsigned int len;
{
  lbit crc;
  int temp;

  crc.l = 0;
  while (len > 0) {
    temp = crc.U1.b0 ^ (*b);
    len--;
    b++;
    crc.U1.b0 = crc.U1.b1;
    crc.U1.b1 = crc.U1.b2;
    crc.U1.b2 = crc.U1.b3;
    crc.U1.b3 = 0;
    crc.l ^= crctable[temp];
  }
  return (crc.l);
}
```

Codifica MD5

Le informazioni su tale tipo di codifica si possono trovare nel documento rfc1321.ps al seguente link: http://rfc.dotsrc.org/rfc/rfc1321.html

Struttura dati per il campo header IP

La struttura dati per il campo header IP (20 bytes) potrebbe essere definita come segue:

```
#define IP_VERSION       4   /* IPv4 Version a quattro bytes */
#define TYPE_UDP        11  /* UDP packet (hexadecimal value = 17d) */
#define TYPE_TCP         6   /* TCP packet */
#define TYPE_ICMP        1   /* ICMP packet */

typedef struct h_ip {
  /* UDP header */

  byte          lenverip;       /* versione IP (1 byte) – da memorizzare */
  byte          typeservice;    /* tipo di servizio ip (1 byte) – da memorizzare */
  word          lenipudp;       /* lunghezza del datagramma (2 bytes) */
  word          idnumber;       /* numero identificativo del pacchetto (2 bytes) */
  word          frouter;        /* per la ricomposizione ad opera dei router (2 bytes) – da memorizzare */
  byte          timetolive;     /* numero massimo dei routers transitabili (1 byte) – da memorizzare */
  byte          type;           /* tipo di pacchetto incapsulato in IP (1 byte) – da memorizzare */
  word          chkip;          /* router cheksum (2 bytes) – da memorizzare */
  longint       sourceipaddr;   /* indirizzo IP sorgente (4 bytes) – da memorizzare */
  longint       destipaddr;     /* indirizzo IP destinazione (4 bytes) – da memorizzare */
} h_ip;

h_ip.lenverip = IP_VERSION;       /* da memorizzare */
h_ip.type = TYPE_UDP;             /* da memorizzare */
```

Struttura dati per il campo header UDP

La struttura dati per il campo header UDP (8 bytes) potrebbe essere definita come segue:

```
typedef struct h_udp {
  /* UDP header */

  word          sourceport;     /* indirizzo (2 bytes) porta UDP sorgente dati – da memorizzare */
  word          destport;       /* indirizzo (2 bytes) porta UDP destinazione dati */
  word          length;         /* lunghezza (2 bytes) (?) */
  word          chksum;         /* checksum (2 bytes) (?) */
} h_udp;
```

Struttura dati per il campo header QDP

La struttura dati per il campo header QDP (12 bytes) potrebbe essere definita come segue:

```
#define QDP_VERSION      2   /* QDP Version for Software Version 1.xx */

typedef struct h_qdp {
  /* common header for all UDP messages */

  longint        crc;             /* calcolato sull'intero pacchetto! */
  byte           command;         /* Comando – come da tabella comandi */
  byte           version;         /* version, da memorizzare */
  word           datalength;      /* senza includere i 40 bytes di header IP, UDP, QDP */
  word           sequence;        /* Sender's sequence */
  word           acknowledge;     /* and acknowledge */
} h_qdp;

h_qdp.version = QDP_VERSION;      /* da memorizzare */
```

Struttura dati per la Registrazione

La struttura dati per la procedura di registrazione potrebbe essere definita come:

```
#define C1_CACK          0xa0      /* Command Acknowledge */
#define C1_RQSRV         0x10      /* Request Server Registration */
#define C1_SRVCH         0xa1      /* Server Challenge */
#define C1_SRVRSP        0x11      /* Server Response */
#define C1_CERR          0xa2      /* Command Error */
#define C1_DSRV          0x12      /* Delete Server */

/* Command Error Codes */
#define CERR_PERM        0         /* No Permission */
#define CERR_TMSERV      1         /* Too many servers */
#define CERR_NOTR        2         /* You are not registered */
#define CERR_INVREG      3         /* Invalid Registration Request */
#define CERR_PAR         4         /* Parameter Error */
#define CERR_SNV         5         /* Structure not valid */
#define CERR_CTRL        6         /* Control Port Only */
#define CERR_SPEC        7         /* Special Port Only */
#define CERR_MEM         8         /* Memory operation already in progress */
#define CERR_CIP         9         /* Calibration in Progress */
#define CERR_DNA        10         /* Data not Available */
#define CERR_DB9        11         /* Console Port Only */

typedef longword t64[2];           /* sixty four bit fields */
typedef longword t128[4];          /* 128 bit fields */

typedef struct tsrvch {
  /* C1_SRVCH */
  t64 challenge;   /* challenge value */
  union {
    t64 md5equiv;
    struct {
      longword dpip;      /* DP IP */
      word dpport;        /* UDP PORT */
      word dpreg;         /* Registration */
    } U1;
  } UU;
} tsrvch;

typedef struct tsrvresp {
  /* C1_SRVRSP */
  t64 serial;             /* serial number */
  t64 challenge;          /* challenge value */
  t64 md5equiv;           /* other information */
  t64 counter_chal;       /* server's counter challege value */
  t128 md5result;
} tsrvresp;
```

Riferimenti

- Q330 Communications Protocol.pdf – Rev. 15
- Q330 DP Writers Guide.pdf – Rev. 4
- rfc1321.ps
- SEED Reference Manual.pdf
- Q330 Response Description.pdf (Guidelines for SEED Data)
- Connector Description for Q330.pdf