TELEDYNE PARADISE DATACOM
A Teledyne Technologies Company

Remote M&C Specification for Q Series Satellite Modems

Issue 3.0.8, 29 April 2013

CE 2013
EN 55022 - Class B
EN 55024
EN 60950

Teledyne Paradise Datacom Ltd.
2&3 The Matchyns, Rivenhall End,
Witham, Essex, CM8 3HA, England.
Tel: +44(0)1376 515636
Fax: +44(0)1376 533764

Teledyne Paradise Datacom LLC
328 Innovation Blvd.
State College, PA 16803, U.S.A.
Tel: +1 814 238 3450
Fax: +1 814 238 3829

http://www.paradisedata.com

Table of Contents

Chapter 1 Introduction

This document specifies the remote control protocols for Teledyne Paradise Datacom (Paradise) Q Series satellite modems. The Q Series covers the Q-Flex dual-band modem and the Q-Lite compact L-band modem.

Several protocols are available:

- A proprietary protocol called Paradise Universal Protocol (PUP). This provides a human-readable, command-response method of controlling equipment that is independent of the underlying physical communications medium. As a result, PUP can be used over serial and Ethernet interfaces. Its capabilities make it suitable for simple Telnet sessions for accessing the modem, as well as when developing fully-automated replacement user interfaces (the built-in modem user interfaces themselves use PUP exclusively as the means for controlling the modem).

- Simple Network Management Protocol (SNMP). The modem includes an SNMP agent. The agent responds to requests from SNMP network managers and also sends SNMP traps notifications to them. The SNMP Management Information Base (MIB) can be downloaded directly from the modem (via the 'Download MIB files' hyperlink at the foot of the *Edit->Unit->M&C->SNMP* web page) or can be obtained from Customer Support. The MIB defines a set of Object Identifiers (OID) that can be used for monitor and control purposes.

- A legacy RS232/RS485 serial protocol that is backwards compatible with older Paradise satellite modems including the Evolution, Quantum and P300 Series modems.

1.1 Related Documents

1. Q Series Installation and Operating Handbook

1.2 Definitions

M&C	Monitor and Control
MCP	Modem Configurable Property
MIB	Management Information Base
OID	Object Identifier
PUP	Paradise Universal Protocol
SAF	Software Activated Feature
SNMP	Simple Network Management System

Chapter 2 Remote Control Overview

As well as the built-in remote web user interface, the modem supports the following remote control interfaces:

- An RS232/RS485 serial interface. This is a nine-way D-type connector labeled 'REMOTE' on the Q-Flex modem rear panel.

  This can be used with the Paradise Universal Protocol (PUP) (described in Chapter 3) or the legacy Paradise serial protocol (described in Chapter 4). The serial interface can be used with a general-purpose command-line application such as HyperTerminal or with a custom-developed application that implements the required protocol.

  When used with RS485, then a message wrapper (defined in Section 2.3.3) must be used to encapsulate PUP commands and responses, which are transported in the message payload.

- An Ethernet interface. This is an RJ45 connector labeled 'ETHERNET M&C' on the Q-Flex modem rear panel.

  This can be used to send and receive PUP or SNMP messages. This interface can be used in several ways.

  First, a general user-entry application such as Telnet can be used to send PUP commands (either manually or automatically).

  Second, under the control of a purpose-developed user application, PUP commands can be encapsulated as the payload of TCP packets. These must be sent to a specific TCP port where they will be recognised and processed by the modem.

  Third, SNMP v1, v2c or v3 can be used to communicate between an SNMP network manager and the SNMP agent on the modem.

Chapter 3 Paradise Universal Protocol

3.1 Command

The Paradise Universal Protocol (PUP) command protocol is human-readable. Since in the simplest scenario it is designed to be used with a command-line application, all commands must be terminated with carriage return/line feed (typically '\r\n' in most computer languages) even when control is automated.

As an example, the following sets the modem Tx service to DVB-S2 (note that carriage return and line feed characters are not explicitly shown):

set TBBTxService DVBS2

Telnet sessions should log in to the modem using the user name *pup* with default password *TEST.* It is strongly recommended that the password is changed.

Other than when using a command-line application, PUP commands must be encapsulated within an RS485 wrapper, even when using Ethernet. RS485-formatted Ethernet PUP commands should be sent to TCP port 6701 for processing by the modem.

When using Ethernet or the RS485 serial interface, PUP commands should be encapsulated as RS485 Write commands (see Section xx for more details). (Telnet and RS232 command-line sessions do not need this encapsulation as it is sufficient to simply present the PUP commands themselves.) For Ethernet, the RS485 Write command should be placed in the payload of a TCP packet and addressed to a specific port on the modem. Several ports are used depending on what action is required. Port usage is described in Section 3.4.

3.2 Response

Every PUP command will generate a corresponding response from the modem. The initiating equipment should wait for each response before sending the next command.

The response will always end with a '$' termination character followed by a carriage return.

If a command fails then the response will be a '!' character followed by an error message and then the '$' and carriage return.

In all cases, the terminating '$' tells the initiator that the modem is ready for the next command.

The modem will send a '$' when the initiating equipment first makes a connection, thereby indicating that it is ready to receive its first command.

When using Ethernet or the RS485 serial interface, PUP responses should be retrieved using RS485 Query commands (see Section xx for more details). (Telnet and RS232

command-line sessions do not need Query commands as modem responses will be provided automatically.) For Ethernet, the RS485 Query command should be placed in the payload of a TCP packet and addressed to a specific port on the modem. Several ports are used depending on what action is required. Port usage is described in Section xx.

3.3 Controlling the Local Modem

Serial Control

CPUSerialMode (shown on the *Edit->Unit->M&C* web page as 'Remote M&C interface') controls the mode of operation of the M&C serial interface. For local modem control this should be set to *RS232* or *RS485*. The modem will then process locally any PUP commands (or legacy RS232/RS485 commands) received on the serial interface.

Ethernet Control

Local modem control via the IP M&C port can be achieved by addressing the PUP commands to the local modem's M&C IP address using TCP port 6701. Local control is unaffected by whether the M&C and IP traffic ports are bridged together or not.

3.4 Controlling the Remote Modem

A remote modem can be controlled either over the Engineering Service Channel (ESC) or over the main traffic channel. The ESC channel is available in *Closed network plus ESC*, *IBS* and *IDR* modes and can be used for either serial or Ethernet control.

When used for serial data, the ESC is a transparent pipe and the modem does nothing to the data other than output it on the ESC connector at the other end of the link. However, the serial M&C interface and the physical ESC interface can be looped back to each other (as explained below) causing the serial M&C interface to effectively be extended over satellite.

Serial Control

RS485 serial PUP commands can be sent to the local modem for transfer to the remote modem over the ESC channel, which needs to be set at the remote modem to loop the commands back to the remote modem's M&C serial interface. Responses sent out of remote modem's M&C serial interface are similarly looped back to the ESC channel and are received back at the local modem where they are output to the local M&C serial interface.

The set up to achieve this is as follows:

- At both the local and remote modems, *CPUSerialMode* (shown on the *Edit->Unit->M&C* web page as 'Remote M&C interface') should be set to *RS485*.
- At the local modem, *BxESCIntfc* (shown on the *Edit->Tx-Rx->Framing->Overhead* web pages as 'ESC interface type') should be set to *RS485 – Local*. This causes whatever is received over the satellite ESC channel to be transmitted out onto the local modem's serial M&C interface, while at the same time, transmitting all incoming M&C messages out over the satellite ESC channel.

- At the remote modem, *BxESCIntfc* (shown on the *Edit->Tx-Rx->Framing->Overhead* web pages as 'ESC interface type') should be set to *RS485 – Remote*. This causes whatever is received over the remote modem's satellite ESC channel to be looped back to the modem's serial M&C interface input and for any responses it generates and outputs via its serial M&C interface to also be transmitted back to the local modem over the ESC channel where they are output via the local modem's serial M&C interface.

Ethernet Control

The main traffic channel can be used for Ethernet control simply by setting the modem to bridge the Ethernet M&C and IP Traffic ports together. M&C IP packets will be mixed together with the normal IP traffic.

If separation of IP M&C and main channel traffic is required, then *BxESCIntfc* (shown on the *Edit->Tx-Rx->Framing->Overhead* web pages as 'ESC interface type') should be set to *IP* in order to restrict M&C data to the ESC channel.

This mode allows IP packets to be sent over the satellite ESC channel. The ESC channel forms part of an Ethernet bridge and it therefore automatically learns which IP packets should be sent over the ESC channel. Packets will therefore be passed to the remote modem and forwarded to the remote Ethernet network as required. IP packets will be interleaved with AUPC messages when AUPC is switched on.

It is also possible for serial M&C commands to be converted to IP for transmission over the ESC channel and then converted back to serial at the remote modem and output on its serial M&C interface. This allows the ESC channel to be shared between any combination of serial commands, IP packets and AUPC. If serial M&C is not required then it is sufficient to simply set *BxESCIntfc* to *IP.*

The modes available that allow serial M&C commands to be converted to IP for forwarding over the ESC channel are as follows:

- When *CPUSerialMode* (shown on the *Edit->Unit->M&C* web page as 'Remote M&C interface') is set to *IP - Local*, the modem takes serial messages input locally and converts these to IP and sends them over the ESC channel.
- When *CPUSerialMode* is set to *IP – Remote*, the modem strips the payload out of IP packets that arrive over the ESC channel addressed to port 6703 and sends the data to the local serial M&C interface with no changes. Incoming M&C serial messages are returned in a similar way.
- When *CPUSerialMode* is set to *IP – Forward to remote*, the modem forwards any IP M&C packets addressed to port 6703 over the ESC channel to the remote modem. Packets received over the ESC channel addressed to port 6703 from the remote modem are forwarded onto the IP M&C port. In conjunction with setting the *IP – Remote* mode at the remote modem, this gives a way of inputting serial commands as IP and converting them back to serial at the remote modem. Note that packets sent to port 6704 are forwarded over the ESC to port 6701 on the remote modem. At the remote modem, packets received over the ESC on port 6701 are treated as local M&C commands for the modem and go no further.

M&C packets for port 6701 on the local modem can be sent at any time when using the ESC channel in IP mode, regardless of the serial M&C interface setting. These commands will be processed by the local modem.

It is also possible to prefix any PUP command with *esc* in order to force the command to go over the ESC channel to the remote modem.

3.5 Optimising Command Bandwidth

Several features are available that help to minimize the bandwidth requirements for M&C.

Each Modem Configurable Property (MCP) has both a name and a number. The numbers can be used to replace the longer textual names with various PUP commands such as *set, get, getisrelevant, getrelevantoptions,* etc. For example, *get TBBTxTerrDataRate* can be replaced with *get 4.* Numbers are assigned to MCPs starting from 0 and in accordance with the order of MCP names returned by the *enumerate* command.

The *commit count* command returns a count of the number of times the modem configuration has changed. If this number is different to the last time it was fetched, then it indicates the configuration has changed and can be used to minimize the rate at which the modem is polled for configuration data. Volatile data (such as Eb/No) is not flagged via *commit count* command since they are potentially changing all the time and therefore these continue to need to be polled for on a regular basis.

3.6 Committing Changes

When PUP *set* commands are sent to the modem to change its operation, they are implemented immediately unless a *nocommit* parameter is appended to the end of the command. If commands are not implemented immediately, then they remain 'pending' until a PUP *commit* command is sent. Not committing changes until they have all been sent can significantly reduce the time taken to configure a modem.

3.7 Nomenclature

In the command descriptions, text shown between angle brackets (< >) should be substituted with the actual required values.

Section headers identify the actual name of each command. Commands and responses are shown in italics for clarity.

3.8 Command Overview

The PUP commands operate on specific named M&C controls referred to as Modem Configurable Properties (MCPs). A full list of the MCPs is provided in the SNMP MIB for the modem, which can be downloaded directly from the modem (via the 'Download MIB files' hyperlink at the foot of the *Edit->Unit->M&C->SNMP* web page). The MIB can also be obtained from Customer Support. The MIB defines a set of Object Identifiers (OIDs) that corresponds directly to the set of MCPs for the modem. The MIB structure corresponds directly to the modem web user interface menu structure and this can be used to determine the OID corresponding to every menu option.

Two PUP commands, namely, the *get* and *set* commands will typically be used more extensively than other commands.

3.9 alarm

Description

This command is used to show status information on system alarms.

The command takes several parameters as follows:

Parameter 1	Parameter 2	Parameter 3	Action
show	*active*	*tx* *rx* *unit*	Returns list of all current active alarms of selected type
show	*all*		Returns status of all alarms
show	*<AlarmName>*		Returns the status of the selected alarm
suppress	*<AlarmName>*		Shows suppression state of selected alarm
suppress	*<AlarmName>*	*on* *off*	Sets selected alarm to suppressed state
path	*tx, rx*		Returns the time for which the selected path has been clear of alarms
clear	*<AlarmName>*		Sets alarm to inactive state

Response

The response format varies with the command options that are selected. Some examples are given below.

Error Messages

Error messages that can be returned are listed below.

Error Message	Description
!Unknown option	The modem did not recognise a parameter name.

Examples

Command:
alarm show active rx

Response:
RxDemodUnlockedAlarm – true, true, false, false, false, Fault

(where:
RxDemodUnlockedAlarm is the alarm name
true, true, false, false, false, Fault represents, respectively, the current state of the alarm, its persistence, user suppressed status, 'other' suppressed status, system

suppressed status and severity. The persistence status is set to 1 when an alarm has occurred and it will stay set in a latched state even after the alarm has cleared until reset by using the *alarm clear all|name* command, when it will be reported as 0. Suppression refers to the ability of the user or system to suppress particular alarms when an alarm indication is not required. With the current Paradise local and remote user interfaces there is no way for the user to suppress alarms therefore the user suppression status can be ignored if using either of these interfaces. Severity indicates whether the alarm is a fault (1) or a warning (0). Note that this response format is used in several of the other command option responses.)

Command:
alarm path tx

Response:
OK for 20.9mins

3.10 board

Description

This command is used to read basic PCB information from the modem, including the build standard.

The command takes several parameters as follows:

Parameter 1	Parameter 2	Action
		Returns list of all currently fitted board types
<BoardTypeName>		Returns list of all attributes for selected board
<BoardTypeName>	*<AttributeName>*	Returns selected attribute value for given board
all		Returns the entire contents of the non-volatile storage. This includes board information as above and calibration data The information is in XML format.

Response

The response format varies with the command options that are selected. Some examples are given below.

Error Messages

Error messages that can be returned are listed below.

Error Message	Description
<BoardTypeName> not found	The modem did not recognise the board type or attribute name.

Examples

Command:
board

Response:
[ControlZone] [LVDS] [RIF] [TIF]

(where the names of fitted assemblies are shown in square brackets.)

Command:
board ControlZone

Response:
[ManufacturerID] [SerialNumber] [MotherboardSerialNumber] [PersistSAF] [MIH]

(where the names of manufacturing attributes are shown in square brackets.)

Command:
board ControlZone SerialNumber

Response:
10500445

Command:
board all

Response:
```
<P3000>
  <DigitalZone>
    <MOD_MB />
  </DigitalZone>
  <TLB>
    <LMX2485 />
    <MOD_CAL_2C00>F=70000000 S=9600 I_DC=-56 Q_DC=-24 I_GAIN=0
Q_GAIN=0</MOD_CAL_2C00>
    …
  </TLB>
  <RLB>
    <LMX2485 />
    <MIL />
    <CALIBRATED>1</CALIBRATED>
    <DEM_CAL_5C00>G=15.26 S=3064 G_I=15.29 S_I=3407</DEM_CAL_5C00>
      …
```

```
      <CALIBRATED>1</CALIBRATED>
    </RLB>
    <HiStab>
      <HiStabCal>152</HiStabCal>
    </HiStab>
    <ControlZone>
      <ManufacturerID>Paradise Datacom</ManufacturerID>
      <SerialNumber>312000037</SerialNumber>
      <MotherboardSerialNumber>Q3248#312000037</MotherboardSerialNumber>
      <ModelNumber>Q-Flex</ModelNumber>
      <ModelType>Q3248</ModelType>
      <PersistSAF>67072058390778113381004319042983 8</PersistSAF>
      <SAF64BitPersist>159817445159502059786286869156693 0</SAF64BitPersist>
      <Chksum>1</Chksum>
      <MIH />
      <SAF64BitPersist1>199605156672648326046393213988401</SAF64BitPersist1>
      <SAFRental />
      <SAFRentalSeq>154671585580591718089413710663409 3</SAFRentalSeq>
    </ControlZone>
  </P3000>
```

3.11 commit

Description

This command is used to reconfigure the modem. It commits all pending MCP changes (i.e. those MCPs that have been modified using the *set* command) to the modem hardware. Use of the *set* command by itself has no impact on the modem. The command can also be used to find out how many times the modem configuration has been changed.

The command takes one parameter as follows:

Parameter	Action
count	This causes the modem to return the number of times the modem configuration has been changed. This can be used to minimize the rate at which the modem is polled for configuration information by detecting when the configuration has changed.

Response

None.

Error Messages

Intentionally blank.

3.12 default

Description

This command is used to set one or all MCPs back to their factory default settings.

The command takes one of two forms:

default all

(which sets all MCPs back to their factory default settings.)

default <MCPName>

(which sets the named MCP back to its factory default setting.)

Response

None.

Error Messages

Intentionally blank.

3.13 demod

Description

This command is used to fetch the data used to create the spectral and constellation web browser graphs.

The command takes one parameter as follows:

Parameter	Action
sym	Returns 512 pairs of data values. The first value is the offset from the current centre frequency and the second value in the signal level in dBm. All values are comma separated.
spect	Returns 1024 pairs of data values. These values represent the x and y co-ordinates of the constellation points. All values are comma separated.

Response

Intentionally blank.

Error Messages

Intentionally blank.

Example

Command:
demod spect

Response:
-1550293,-78,-1544189,-74,-1538086,-73,-1531982,-72, ……

Command:
demod sym

Response:
-59,58,-68,-60,59,-64,-62,-63,58,-63,-63,-62,64,-63,67,65,-65,67,-72,……..

3.14 enumerate

Description

This command is used to get the name of every MCP property supported by the modem software.

The command does not take any parameters.

Response

TBBTxService
TBBTxServiceStrict
TBBTxFlexFrmIDR
etc.

Error Messages

Intentionally blank.

3.15 esc

Description

This command can be used to prefix any other PUP command documented here. If a command is prefixed with esc then it is passed over the ESC channel and executed on the remote modem. Any response is transmitted back to the initiating modem. The ESC channel must be setup and functioning correctly. As the data rate of the ESC channel is considerably less than the main channel PUP commands executed remotely may take a noticeable time to respond. Also these commands will be affected by the round-trip delay over the satellite link.

If a command is issued that breaks the ESC link (i.e. changes the configuration of the link) no response will be forthcoming and no further esc commands can be issued.

To check the status of the link the parameter *linkstatus* can be used, the command will then return either *OK* if the ESC is operational or *Failed* if it is not.

Response

The response is the same as if the command had been executed locally.

Error Messages

Error Message	Description
!Unable to connect	The modem ESC channel is not established

Example

Command:
esc get GwyTxCarrier

Response:
On

3.16 framer

Description
This command calculates and displays the current percentage of the signal bandwidth that is occupied by the overhead channel.

The command takes one parameter as follows:

Parameter	Action
txoverhead	Returns the percentage of the Tx signal bandwidth that is occupied by the overhead channel
rxoverhead	Returns the percentage of the Rx signal bandwidth that is occupied by the overhead channel

Response

See the example below.

Error Messages

Intentionally blank.

Example

Command:
framer txoverhead

Response:
7%

3.17 get

Description

This command fetches the current value of an MCP. The MCPs supported by the modem are defined in Chapter 5.

The command takes one or more parameters separated by a space as follows:

Parameter	Action
<MCPName> [<MCPName>]	Returns the current values of the selected MCPs

Response

See the example below.

Error Messages

Error messages that can be returned are listed below.

Error Message	Description
!Unknown variable name. Please correct & resubmit.	The modem did not recognise the MCP name.
!Syntax Error.	The modem received a message it does not understand.

Example

Command:
get GwyTxCarrier

Response:
On

Command:
get GwyTxCarrier TIFTxIFFreq

Response:
On,81

3.18 getattrib

Description

This command allows the initiating equipment to request the minimum value, maximum value, units and step size associated with a particular MCP.

The command takes several parameters as follows:

Parameter 1	Parameter 2	Action
min	*<MCPName>*	Returns the minimum valid value for the selected MCP
max	*<MCPName>*	Returns the maximum valid value for the selected MCP
units	*<MCPName>*	Returns the units for the selected MCP
step	*<MCPName>*	Returns the smallest increment in size that the selected MCP can be changed by

Response

See the examples below.

Error Messages

Intentionally blank.

Examples

Command:
getattrib min RRSRxRSN

Response:
58

Command:
getattrib max RRSRxRSN

Response:
255

Command:
getattrib units RRSRxRSN

Response:
symbols

Command:
getattrib step RRSRxRSN

Response:
1

3.19 getcurrent

Description
This command returns a snapshot of the current configuration of the modem. It is similar to the *getcurrentconfig* command but does not compress the data and it returns only the actual current configuration (without alarm information). In addition, to make the data smaller, the name of each MCP is replaced by a number that uniquely identifies each MCP. The number abbreviations are explained in Section xx.

Note that the command returns only the differences from the factory default settings. The factory default setting is listed for each MCP in Section 5.

The command does not take any parameters.

Response

The command responds with a text string listing all MCPs that have changed from their factory default settings and gives their current values. MCP names are replaced with unique numbers to minimize the size of the response.

Error Messages

Intentionally blank.

Example

Command:
getcurrent

Response:
230=PEP
232=Off
238=TakeAway
245=193.25.15.1
247=193.251.150.116
246=255.255.255.0
257=Ignore
235=193.251.150.115
236=255.255.255.0
255=Ignore
90=On
201=0
172=K32x2
217=Turbo
92=Closed
96=10000000
204=QAM16
209=TPC
210=R14280_16320
219=144
518=0.95

87=Turbo
0=Closed
4=10000000
79=TPC
80=R14280_16320
88=144
89=-25
517=0.95
76=QAM16
222=IP
298=
299=
$

3.20 getcurrentconfig

Description
This command returns a snapshot of the current configuration of the modem plus alarm information in a compressed form.

The command does not take any parameters.

Response

The command responds with a uuencoded, gzip'd tar archive containing 4 files:

default.conf - Configuration memory containing the modems current configuration
tmp/alarmrx.conf - Active or latched Rx alarms as returned by the alarm command
tmp/alarmtx.conf - Active or latched Tx alarms as returned by the alarm command
tmp/alarmunit.conf - Active or latched Unit alarms as returned by the alarm command

Error Messages

Intentionally blank.

Example

Command:
getcurrentconfig

Response:

```
begin 644 conf.tar.gz
M'XL(``````"`^V2W6J#,!A`O>Y3B`]@$XT5H2NH^T&VL6+M`P2-16;,FL;6
M[NEGM]664==UK+)"SHV2G"\$3@1]Z>,<<UH6F=!C5J3*GP,@``.$%/#!U^_[
M/P0((<N`P+!KWS2`I:A`Z8!R(3!7584S)K[SCNU?*&+;GU=GJO^;_L;`LF7_
M3ON+_]5_`&7_+DA(BLM<G*W]#_I#8)O;_@"9:-,?6+)_)PPI2PC=U,]F:H$I
MN=+VGX0VZJDUPP41G[O^>!I.@S%G@L4LU]0ESLMZ.<+/Q%WAM:;V#XX0ZK,B
M&+M)PIL9"'2H0\?1';MMK'*#B9M3-Q;-5#`K&">M`S?SJ&K<IZ+%BTXZ^&ZU
MCBH?<YX1?N3LT//"RBO3E/!)]DH:&[3*P77HEDG&'NL0C7YO&I71.C(A?)G%
M.]O/V8(DA_3(\Z)3](=:]Z;^+2?SW=5UQWJ7A_V]QS+J*1*)1"*Y7-X`1+\J
%E``0````
`
```

```
end
```

3.21 gethelptext

Description
This command allows the initiating equipment to request the 'tooltip' Help text associated with a particular MCP.

The command takes one parameter as follows:

Parameter	Action
<MCPName>	Returns the Help text associated with the selected MCP

Response

See the example below.

Error Messages

Intentionally blank.

Example

Command:
gethelptext TBBTxService

Response:
Framing mode for the Tx path.

3.22 getisrelevant

Description

This command allows the initiating equipment to request the 'relevance' of a particular MCP.

Relevance is typically used to decide whether to 'gray out' or hide options that are not currently available to the user. An MCP may be irrelevant due to the hardware/software build standard of the modem not supporting the feature, or, because the modem is in a mode of operation where the MCP is not used (for example, all transmit properties are irrelevant when the transmit service is switched off).

The command takes one parameter as follows:

Parameter	Action
<MCPName>	Returns *true* if the selected MCP i relevant and *false* otherwise

Response

true or *false*

Error Messages

Error messages that can be returned are listed below.

Error Message	Description
!No such variable	The modem did not recognise the MCP name.

Example

Command:
getisrelevant TBBTxService

Response:
true

3.23 getisvalid

Description

This command allows the initiating equipment to request the 'validity' of a particular MCP.

An MCP is valid if its current value is within the range of values that allow correct system operation. This may be a subset of the overall set of values that an MCP can take, with the subset being determined by other operational settings. For example, a FEC rate of 0.667 is valid if the FEC mode is TCM but not when Viterbi is selected. Validity is typically used to identify configuration errors. Note that it differs from MCP relevance in that an MCP may be relevant but invalid and vice versa.

The command takes one parameter as follows:

Parameter	Action
<MCPName>	Returns *true* if the selected MCP i relevant and *false* otherwise

Response

true or *false*

Error Messages

Error messages that can be returned are listed below.

Error Message	Description
!Variable does not exist	The modem did not recognise the MCP name.

Example

Command:
getisvalid TBBTxService

Response:
true

3.24 getlabel

Description

This command allows the initiating equipment to request the predefined display label associated with a particular MCP.

This returns the display label that is shown alongside the current value of a particular MCP that is used to identify it to the user. It is typically used when building a user interface. The text that is returned is that which is used on the web user interface rather than the abbreviated form of this text that is used on the local user interface.

The command takes one parameter as follows:

Parameter	Action
<MCPName>	Returns the display label associated with the selected MCP

Response

See the example below.

Error Messages

Error messages that can be returned are listed below.

Error Message	Description
!Unknown variable name. Please correct & resubmit.	The modem did not recognise the MCP name.

Example

Command:
getlabel RBBRxClkMode

Response:
Rx-path clock source

3.25 getoptions

Description

This command allows the initiating equipment to request the options associated with a particular MCP (such as the list of FEC rates or modulation schemes supported by the modem).

This returns both the internal value of each option and the display text shown to the user. The internal values are those that are used with the *set* command to change system operation. This command is typically used when building a user interface. The text that is returned is that which is used on the web user interface rather than the abbreviated form of this text that is used on the local user interface.

The command takes one parameter as follows:

Parameter	Action
<MCPName>	Returns a list of option values and associated display labels for the selected MCP

Response

Returns a list of *<option value> <label string>* pairs. The first space character terminates the option value. The label string may contain spaces and is terminated by <cr><lf> (carriage return/line feed).

Error Messages

Error messages that can be returned are listed below.

Error Message	Description
!Variable name does not exist	The modem did not recognise the MCP name.

Example

Command:
getoptions RBBRxService

Response:
Off Off
Closed Closed network
MinOH Closed network plus ESC
IBSSMS IBS/SMS
IDRIDR
OM73 OM-73

(where the first word in each line is the option name and the remainder of each line is the display text, for example, *MinOH* is the value that *RBBRxService* must be set to in order to enable the Closed network plus ESC service.)

3.26 getreadonly

Description

This command allows the initiating equipment to determine if a particular MCP can be written. Some MCP's are designated read-only as they contain values that can dynamically change (such as RxEbNo) or are not directly set by the user (such as RxSymRate).

The command takes one parameter as follows:

Parameter	Action
<MCPName>	Returns *true* if the selected MCP i read-only and *false* otherwise

Response

true or *false*

Error Messages

Error messages that can be returned are listed below.

Error Message	Description
Unknown variable name. Please correct & resubmit.	The modem did not recognise the MCP name.

Example

Command:
getreadonly TBBTxService

Response:
false

3.27 getrelevantoptions (abbreviation: gro)

Description

This command allows the initiating equipment to request only those options associated with a particular MCP that are currently relevant to system operation.

This returns a subset of the option information returned by the *getoptions* command. It is typically used when building a user interface to present only those options to a user that are currently relevant (i.e. selectable).

The command takes one parameter as follows:

Parameter	Action
<MCPName>	Returns a list of relevant option values and associated display labels for the selected MCP

Response

Returns a list of *<option value> <label string>* pairs. The first space character terminates the option value. The label string may contain spaces and is terminated by <cr><lf> (carriage return/line feed).

Error Messages

Error messages that can be returned are listed below.

Error Message	Description
!Variable name does not exist	The modem did not recognise the MCP name.

Example

Command:
getoptions RBBRxClkMode

Response:
Sat Satellite
Tx Tx Clock In
Int Internal
RxRef Receive reference

(where the first word in each line is the option name and the remainder of each line is the display text – in this example, no option information is returned for station clock because it is irrelevant, typically due to no clock source for station clock having been selected.)

3.28 gettype

Description

This command allows the initiating equipment to request the type associated with a particular MCP.

The returned value can be used to determine how to present the MCP to a user when building a user interface, for example, whether to display a drop-down box with a list of options or to display an edit box that takes a numeric value.

The command takes one parameter as follows:

Parameter	Action

<MCPName>	Returns the type of the selected MCP

Response

Returns a single value indicating the type of the MCP variable. This will be one of the following:

- *Range*
- *Group*
- *Text*
- *Float*
- *Bool*
- *DottedDecimal*
- *Alarm*

where:

- *Range* indicates the MCP is a numeric integer value (e.g. TBBTxTerrDataRate)
- *Group* indicates the MCP takes one of a set of specific values (e.g. TBBTxService)
- *Text* indicates the MCP is a text string (e.g. CPURUIPassword)
- *Float* indicates the MCP is a floating point value (e.g. RxFinalBER)
- *Bool* indicates the MCP is a Boolean variable (e.g. TFECTxSpectInv)
- *DottedDecimal* indicates the MCP is a IP address or subnet mask (e.g. CPURemConIPAddr)
- *Alarm* indicates the variable is a read-only text string representing an alarm description.

Error Messages

Error messages that can be returned are listed below.

Error Message	Description
!Unknown variable type	The modem did not recognise the MCP name.

Example

Command:
gettype RBBRxClkMode

Response:
Group

3.29 help

Description

This command returns a list of most but not all PUP commands.

Some commands function at an access control level (rather than command handler level) and will not appear on the list returned by the *help* command – use this manual as a definitive guide to what is supported. Other commands that are listed by the *help*

command are **not** listed in this manual – this is because they are reserved for internal use by Paradise Datacom (note that use of these reserved commands may result in unpredictable system behaviour).

The command can optionally take one parameter as follows:

Parameter	Action
<PUPCommandName>	Returns usage information associated with the selected command (returns information on all commands if no parameter is supplied)

Response

See example below.

Error Messages

Intentionally blank.

Example

Command:
help gettype

Response:
gettype variable name
Returns the type of the given variable

Description

This command returns a list of all PUP commands.

The command can optionally take one parameter as follows:

Parameter	Action
<PUPCommandName>	Returns usage information associated with the selected command (returns information on all commands if no parameter is supplied)

Response

See example below.

Error Messages

Intentionally blank.

Example

Command:
help gettype

Response:
gettype variable name
Returns the type of the given variable

3.30 incontrol

Description

This command is a query that can be used to indicate whether the initiating equipment has control of the target equipment. Control refers to the ability to be able to change the modem configuration. Only one user can be in control of the modem at any one time. The rules governing modem control are specified by the MCP *CPURUIProtocol*.

This command does not take any parameters.

Response

true or *false.*

Error Messages

Intentionally blank.

Example

Command:
incontrol

Response:
false

3.31 load

Description

This command is used to load a configuration memory into the modem's operational settings.

Configuration memories are used to store specific sets of operational settings for quick recall. The *load* command is used to recall previously stored configuration memories.

The command takes one parameter as follows:

Parameter	Action
<ConfigurationMemoryName>	Fetches the selected configuration memory and applies it to the modem hardware, replacing the current operational settings

Response

None.

Error Messages

Intentionally blank.

Example

Command:
load QPSKVitRate12

Response:
Intentionally blank.

3.32 log

Description

This command is used to display and clear the system log.

Parameter	Action
show	Lists all the entries currently in the log
clear	Deletes all the entries currently in the log
auto	Reserved for future use

Response

None.

Error Messages

Intentionally blank.

Example

Command:
log show

Response:
Apr 11 08:44:10 (none) user.info P300[123]: mcp.xml loaded OK
Apr 11 08:44:15 (none) user.info P300[123]: RxDemodUnlockedAlarm Raised
Apr 11 08:44:18 (none) user.info P300[123]: Not starting SNMP as variable RunSNMP is not set
Apr 11 08:45:32 (none) user.debug P300[118]: starting up pupclient server

3.33 login

Description

This command is used to log in to the modem software application. A brief description of the concepts involved is given below.

The modem has both front-panel and web browser user interfaces.

For remote web browsing, there are two fixed user names, namely, *admin* and *user*. The *admin* user can view and change the modem configuration, while *user* can only view the modem settings. Only *admin* can change the passwords associated with these two user names.

There is no restriction on the number of users (as either *admin* or *user*) that can be logged in at the same time. Remote *admin* users who log in while the modem is under local front-panel control will be restricted to view-only permissions.

Local Mode

Local mode allows control of the modem from the front-panel interface only. Web users are still able to log in and view the modem settings in this mode.

Takeaway Mode

In *Takeaway mode*, the modem can be controlled through the front-panel or via a remote *admin* user at the same time. When the modem is switched out of *Takeaway mode* to *Local mode* then all remote *admin* users will be automatically logged out.

While *Takeaway mode* is very convenient, it is essential for there to be clear operational procedures in place to avoid conflicts arising in relation to modem control.

The command takes a single parameter, namely, a password.

If unsuccessful, the initiating equipment may still successfully issue commands that do not change the target equipment configuration.

This command takes one parameter as follows:

Parameter 2	Action
<Password>	Attempts to log the user in to the modem software application as the given user with the given password

Please note that the need to provide a user name applies to the web user interface only – the underlying PUP protocol requires a login only for the admin user in order to take control of the modem, i.e. any PUP session has view-only permission by default.

Response

None.

Error Messages

Error messages that can be returned are listed below.

Error Message	Description
!You must enter a password to login.	An attempt was made to log in without providing a password.
!User is not logged in. Please re-enter password.	An attempt was made to log in using an invalid password.

Example

Command:
login admin myPassword

Response:
Intentionally blank.

3.34 logout

This command logs the user out of the modem software application.

Response

None.

Error Messages

Intentionally blank.

Example

Command:
logout

Response:
Intentionally blank.

3.35 monitor

Description

This command allows up to one month of time-based performance data to be fetched from the modem. Data is automatically stored from power up for each of the modem's performance web graphs (such as Eb/No, Rx power level, modem temperature, etc.). It is also possible to customize the list of parameters for which data is dynamically monitored by adding to or subtracting from the list (although this does not change the web graph pages themselves).

The command takes up to three parameters as follows:

Parameter 1	Parameter 2	Parameter 3	Action
list			Returns a list of the modem parameters that performance data is currently being measured and stored for
add	*<MCPName>*		Adds the given modem parameter to a list that the modem automatically monitors on a regular basis and stores measured values for for up to 31 days
remove	*<MCPName>*		Removes the given modem parameter from the list of monitored parameters
get	*minute* *hour* *day* *month*	*<MCPName>*	Retrieve the data of the specified duration for the given modem parameter

Response

See below.

Error Messages

Error messages that can be returned are as follows.

Error Message	Description
Cannot monitor variable <MCPName>	The modem did not recognise the name of the parameter to be monitored or the parameter is not of a type that changes value dynamically.
Cannot remove variable <MCPName>	The modem did not recognise the name of the parameter to be removed from the dynamic monitor function.
Time period must either be minute,hour,day or month	The time period entered with the *monitor get* command was not recognized.

Example

Command:
monitor get day RxPwrLevel

Response:

```
-27.596 -28.6227 -26.9937
-27.5996 -28.4998 -27.0093
-27.6045 -28.6392 -26.9914
-27.5997 -28.5098 -27.1
-27.6013 -28.5996 -26.7649
-27.6041 -28.6298 -26.8727
-27.5943 -28.5696 -26.8005
-27.6007 -28.7035 -26.8965
-27.594 -28.5687 -26.8033
-27.6003 -28.5827 -27.0089
-27.5922 -28.5728 -26.9884
-27.5947 -28.6123 -26.8969
-27.5968 -28.7311 -26.977
-27.5888 -28.5658 -27.0065
-27.5946 -28.6625 -26.9111
-27.595 -28.5545 -26.9189
-27.5879 -28.5847 -26.9439
-27.5941 -28.5487 -26.9593
-27.5882 -28.5759 -26.9423
-27.5889 -28.5484 -26.8109
-27.603 -28.5149 -26.9706
-27.5985 -28.1791 -27.1802
-27.6001 -28.4018 -27.1105
-27.596 -28.1443 -27.0443
$
```

3.36 oneforone

This command is used to control the modem when it is being used in a one-for-one redundancy configuration. Note that this command cannot be used to switch a modem from standby to on-traffic.

Parameter	Action
switch	Switches the modem from on-traffic (*main*) to being the standby (*standby*) modem in a 1-for-1 redundancy pair
status	Returns the current status of the modem

Response

See the example below.

Error Messages

Intentionally blank.

Example

Command:
oneforone status

Response:
main

3.37 ping

Description

This command is used to send an ICMP *ping* command from the modem to a given IP address.

This command takes one parameter as follows:

Parameter	Action
<IPAddress>	Issues an ICMP *ping* command from the modem to the given IP address and returns the result

Response

See the example below.

Error Messages

Error messages will be those generated by the standard ICMP *ping* command.

Example

Command:
ping 10.0.20.12

Response:
PING 10.0.20.12 (10.0.20.12) : 56 data bytes
--- 10.0.20.12 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/mdev = 0.216/0.231/0.240/0.018 ms

3.38 prbs

Description

This command is used to control the built-in PRBS Bit Error Rate generator and fetch the results.

Parameter	Action
reset	Clears the bit error count and sets the elapsed test time to zero
inject	Injects one bit error into the transmitted stream
sync	Returns the current sync status (i.e. *OK* or *NO SYNC* - indicating whether synchronisation with the test pattern is being maintained)
ber	Returns the bit error rate (this equals the result of dividing the number of errors by the number of bits received – see the next two parameters)
errors	Returns the number of errors since the test started
bits	Returns the number of bits received since the last synchronisation with the test pattern occurred
loss	Returns the number of times the sync has been lost since the test started
time	Returns the elapsed time since the start of the test

Response

Intentionally blank.

Error Messages

Intentionally blank.

Example

Command:
prbs sync

Response:
NO SYNC

Command:
prbs ber

Response:
<3.1E-08

Command:
prbs time

Response:
29.4mins

3.39 reconfig

Description

This command is used to query the IP settings of the modem. Its primary use is when the modem is configured to obtain its IP address using DHCP.

Parameter	Action
addr	Returns the modems currently assigned IP address
mask	Returns the modems currently assigned netmask
gway	Returns the modems currently assigned default gateway address

Response

Each of the above parameters will return an IP address in the form a.b.c.d.

Error Messages

If the modem does not currently have an assigned address the command will respond with an empty string.

If the modem has an assigned address but no assigned default gateway the *gway* option will respond with 0.0.0.0.

Example

Command:
reconfig addr

Response:
10.0.70.1

3.40 reset

Description

This command is used to reboot the modem.

This command does not take any parameters.

Response

There is no response since the target equipment will close down the software and restart.

Error Messages

Intentionally blank.

Example

Command:
reset

Response:
Intentionally blank.

3.41 save

Description

This command is used to save the current operational modem settings to a named configuration memory.

This command takes one parameter as follows:

Parameter	Action
<ConfigurationMemoryName>	Stores the current operational settings into the selected configuration memory

Response

The message '*File saved OK.'* will be displayed if the save was successful.

Error Messages

Intentionally blank.

Example

Command:
save memory1

Response:
Intentionally blank.

3.42 sessionid

Description

This command is a query that returns the unique session identifier associated with the current user login session. It is typically used in conjunction with the *sessions* command.

This command does not take any parameters.

Response

The unique login session identifier for the user issuing this command. These are random numbers that are generated by the modem software to associate different requests from a single user over time in order to provide user login session control.

Error Messages

Intentionally blank.

Example

Command:
sessionid

Response:
274079072

3.43 sessions

Description

This command is a query that identifies which users are currently logged on.

This command does not take any parameters.

Response

A list of all the users that are currently logged on. Each user session is identified by a unique number as shown in the example. The local user interface is identified as *LUI*.

Error Messages

Intentionally blank.

Example

Command:
sessions

Response:

ID	*Logged In*	*Expires*
*LUI**	*yes*	*13:11:22 - 24/4/2004*
274079072	*yes*	*13:13:50 - 24/4/2004*

There are 2 current sessions in total

3.44 set

Description

The set command is used to set a configurable property on the target equipment. The new setting is applied immediately to the target equipment unless the optional parameter *nocommit* is appended. The new setting will then remain 'pending' until a *commit* command is issued that applies the pending changes to the modem hardware.

Chapter 5 defines the modem configurable properties.

This command takes two parameters as follows:

Parameter 1	Parameter 2	Parameter 3	Action
<MCPName>	*<Value>*	nocommit	Sets the current value of the selected MCP to the given value

Response

None.

Error Messages

Intentionally blank.

Example

Command:
set TBBTxService Closed

Response:
Intentionally blank.

Command:
set TBBTxTerrDataRate 2048000 nocommit

Response:
Intentionally blank.

3.45 snmp

Description

This command is used to enable the SNMP agent in the modem. Note that there are various SNMP configuration properties (listed in Chapter 5) that can be controlled using the *set* command. Note that the configuration property *RunSNMP* must be set to *true* before attempting to start SNMP.

This command takes one parameter as follows:

Parameter	Action
start	Starts the SNMP agent in the modem
stop	Reserved for future use
reconfig	Reserved for future use

Response

None.

Error Messages

Intentionally blank.

Example

Command:
snmp start

Response:
Intentionally blank.

3.46 switch

This command is used to get the status of, and control, a one-for-n redundancy switch. A one-for-n system can contain up to 16 modems. The value of n in the following commands is an integer between 1 and 16.

Parameter 1	Parameter 2	Action
status	*<n>*	Returns the status of traffic modem n
status	*All*	Returns the status of all the modems
backup	*<n>*	Forces modem n to be backed-up by placing it into standby
mask	*<n>*	Masks the status of traffic modem n to allow it to be taken offline without causing a changeover
poll		Forces the switch to poll all traffic modems for their current configurations, which are stored by the switch and used in the event of having to back up one of the modems (note that configurations are polled in the background at all times anyway, no less than once per hour)
unmask	*<n>*	Removes the mask set up above
reset		Sets the switch to standby mode

Response

The response can be any of the following:

Response	Meaning
Masked – BackingUp	The modem is being backed up and is also masked (meaning it can be removed for maintenance without reporting its status as *CommsFailure*)
Masked	The modem is masked and cannot cause a switchover to occur
NotFitted	No modem is fitted to channel n on the switch
BackingUp – Standby	The modem is being backed up due to being placed into the standby state
BackingUp – Failed	The modem is being backed up due to a fault occurring in the modem
Failed	The modem has failed but is not being backed up by the switch
Standby	The modem is in a standby state and is not passing traffic
CommsFailure	No response was received by the switch from modem n when it was polled
OK	The modem is not indicating any fault

Error Messages

Intentionally blank.

Example

Command:
switch status 1

Response:
OK

3.47 terr

Description

This command is used to perform real-time control the modems terrestrial interface.

This command can take one parameter as follows:

Parameter	Action
centre	Manually recentre the receive path Doppler buffer

Response

None.

Error Messages

Intentionally blank.

Example

Command:
terr centre

Response:
Intentionally blank

3.48 time

Description

This command is used to retrieve or set the time and date on the modem.

This command takes two parameters as follows:

Parameter 1 (Time)	Parameter 2 (Date)	Action
<HH:MM:SS>	*<DD/MM/YYYY>*	When parameters are provided, sets the modem time and date to the given values, otherwise returns the current time and date in the modem

Response

Returns the current time and date of the modem in the format *HH:MM:SS DD/MM/YYYY*.

Error Messages

It should be noted that changing the modem time may cause a *Giveaway* timeout, resulting in the current control session being terminated.

Example

Command:
time

Response:
11:03:10 25/11/2004

Command:
time 10:00:00 26/02/2005

Response:
10:00:00 26/02/2005

Chapter 4 Serial RS485 Protocol

The Paradise RS485 protocol is compatible with all previous Paradise satellite modems. It was introduced on the P300 Series of satellite modems and was subsequently supporte on Evolution and Quantum modems. As a legacy protocol, the message format and general capabilities remain identical to that used in older modems such as the Evolution and Quantum satellite modems. Newer features of Q Series modems cannot be accessed using the RS485 protocol.

As well as being a protocol in its own right, the RS485 protocol is used to wrapper PUP commands (which give full access to all Q Series modem features). The RS485 wrapper provides a convenient way of making the transmission of PUP commands more robust.

For new installations or for communicating with Evolution and Quantum modems, the PUP protocol should be used in preference to the original P300 RS485 protocol, which has limited capabilities and does not support all of the features of Q Series modems. All Q Series features can be fully controlled by restricting use of the RS485 protocol to Write and Query commands and using the RS485 message payload purely for PUP commands and responses.

The Paradise RS485 protocol is a master-slave, command-response protocol. The master device initiates all communications and slave devices only ever send a message in response to a request from the master.

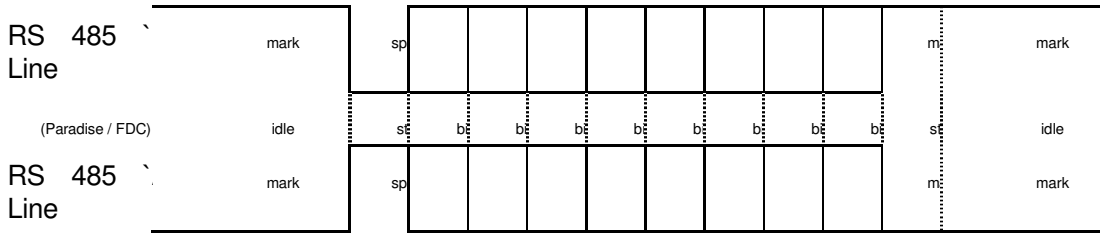
4.1 Character Format/Baud Rate

The character format and Baud rate are as follows: 8 bits, no parity, 1 stop bit, selectable from 300 to 19200 Baud.

4.2 Electrical Interface

The protocol requires a 4-wire plus ground interconnection between equipment. Signals are at RS485 levels (effectively a tri-statable RS422) with Tx & Rx data being transmitted as a series of asynchronous characters over a differential pair (labelled `A` and `B`). Lines referred to as Tx-A and Tx-B are outputs, and Rx-A and Rx-B are inputs. The Paradise convention (as specified by RS422) is that the `B` lines represent true data (i.e. the inactive state is `mark`, which is high), and the `A` lines inverse data (ie the inactive state is `space`, which is low).

An asynchronous character then appears as:

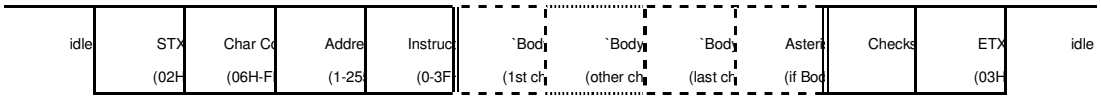


The differential pair from Master to Slaves is typically driven all the time by the Master device (i.e. it never goes high impedance). The return pair (from Slaves to Master) is driven by a Slave device when sending a message. All devices hold their output drivers at a high impedance unless actually transmitting a message, to allow other Slave devices on the bus access to the return signals.

RS232 can be used when there is a single device to be controlled. Note that on RS232 systems, a `mark` (high) is defined as a negative voltage and a space (low) is defined as a positive voltage. This means that an RS232 character will appear as the RS485 `A` line shown previously, except it will transition from <-3V to >+3V as opposed to 0V & 5V.

4.3 Message Structure

All messages are transmitted in the following format, which is the same for Master-to-Slave and Slave-to-Master communications. Messages from Masters carry the address of the destination Slave device. Return messages from Slaves also carry the address of the Slave



STX:	The fixed character 02H.
Char Count:	The message length including STX and ETX characters (06H for zero `Body` length).
Address:	Slave address, range 1-255. Zero is reserved for the `Global` address to which all devices respond.
Instruction:	Range 0-3FH. Add 40H (64_{10}) to request a standard `ACK`, add 80H (128_{10}) to request an `extended ACK` to this message.
Body:	From 0 to 248 characters (resulting in a maximum message length of 255 characters). The body always contains a PUP command as specified in Chapter 4.
Asterisk:	An ASCII `*` character if there is a body (i.e. if the length of the body of the message <>0).
Checksum:	The Modulo 256 sum of all the characters inclusive from the `Address` to the end of the `Body`, up to and including the asterisk.
ETX:	The fixed character 03H.

The following instruction codes are supported:

Message Name	Instruction Code	Description
Ping	63 (3FH)	This results in the command being echoed back by the addressed modem (assuming an ACK or extended ACK has been requested and it is not a broadcast command).
Write	15 (FH)	This is used to send any PUP command (typically a get or set command) as the message body. Any response must be retrieved using the Query command.
Query	14 (EH)	This returns a response (with Instruction Code 14) that is dependent on the last Write command. The body of the message will typically contain the response to the last PUP command. If the reponse exceeds 255 bytes, it will be split into multiple packets, each a maximum of 255 bytes.
Write & Query	9	This combines the Write & Query messages, so that a Write message is sent to the modem, which then responds with a Query response. Both messages have instruction code of 9.

4.4 P300, Evolution and Quantum Modem Compatibility

A set of RS485 messages provide backwards compatability with P300, Evolution and Quantum modems. Not all of the original P300 messages are supported. Those not listed will be acknowledged if they are used, but will not be actioned by the Q Series modem. The degree of backwards compatibility with the P300 in this respect is unchanged from that supported by Evolution and Quantum modems.

Fields shown as ~~strikethrough~~ are not supported on Q Series modems and will be blank when read and ignored when written. For more detailed information regarding these messages see *Remote M&C Specification for P300 & P400 Series Satellite Modems*.

4.5 Message Definitions

The set of Paradise RS485 protocol messages are defined in the following set of tables.

MSG: QFaultRd																	
{Unit Fault active (BxUnitFault_U<>0)}	0-1	1	0=Normal, 1=Active / true	0						X							
{Unit Warn active (BxUnitWarn_U<>0)}	0-1	1	0=Normal, 1=Active / true						X								
{Tx Traffic fault active (TxTrafFault_U<>0)}	0-1	1	0=Normal, 1=Active / true					X									
{Tx Traffic warning active (TxTrafWarn_U<>0)}	0-1	1	0=Normal, 1=Active / true				X										
{Rx Traffic Fault active (RxTrafFault_U<>0)}	0-1	1	0=Normal, 1=Active / true			X											
{Rx Traffic warning active (RxTrafWarn_U<>0)}	0-1	1	0=Normal, 1=Active / true		X												
{Config Error active (BxConfigError<>0)}	0-1	1	0=Normal, 1=Active / true														X
{Config Warn active (BxConfigWarn<>0)}	0-1	1	0=Normal, 1=Active / true													X	
~~{Config Info active (BxConfigInfo<>0)}~~	0-1	1	0=Normal, 1=Active / true												X		
* RemContMode	0-1	1	direct											X			
{Test or Loop active (BxTestMode<>0 OR BxLoopMode<>0)}	0-1	1	0=Normal, 1=Active / true										X				
{On-line / standby, carrier off for any reason (TxCarrierStat_U <> 0)}	0-1	1	0=Normal, 1=Active / true								X						
~~{Local configuration change (cleared when *any* config read remotely)} (Note 7~~ ~~Error! Bookmark not defined.~~ nya)	0-1	1	0=Normal, 1=Active / true	2						X							
~~{UnreadEntries}~~	0-1	1	1=At least 1 unread log entry						X								
~~{Unread16Entries}~~	0-1	1	1=At least 16 unread log entries					X									
{Device configuring / busy}	0-1	1	0=Normal, 1=busy/reconfiguring				X										
(Spare)		1			--												
(Spare)		7			--						--	--	--	--	--	--	
TOTAL		**24**		**4**													

MSG: RxBBRd & RxBBWr																	
RxAsyncBaud	0-15	4	direct (index to Baud array, see appendix D8)	0			msb .. lsb										
RxIBSAuxDat	**0-2**	2	direct		msb	lsb											
RxAsyncChar	0-7	3	direct											msb .. lsb			
RxClkMode	**1-5**	3	direct								msb .. lsb						
RxESCLvlCh1	+7 to -18 dBm (**0-250**)	8	(Lvl x 10) + 180	2	msb .. .. lsb												
RxBBDataRate	0-9,999,999	28	7 digit BCD										msb .. digit 6 .. lsb				
				4	msb .. digit 5 .. lsb				msb .. digit 4 .. lsb				msb .. digit 3 .. lsb				
				6	msb .. digit 2 .. lsb				msb .. digit 1 .. lsb				msb .. digit 0 .. lsb				
RxBBMode	**0-23**	5	direct	8								msb .. lsb					
RxBufferSize	**0-99**	7	direct		msb .. lsb												
RxCustScr	As text	5	direct	10								msb .. lsb					
* RxEqTx	0-1	1	direct								X						
RxIBSCust	0-7	3	direct				msb .. lsb										
RxFrmMode	0-7	3	direct		msb .. lsb												
RxESCLvlCh2	+7 to -18 dBm (**0-250**)	8	(Lvl x 10) + 180	12	msb .. lsb												
RxIDRAuxDat	0-3	2	direct												msb	lsb	
RxIBSESCDat	0-3	2	direct										msb	lsb			

														b				
~~RxIBSAuxOh~~	0-31	5	direct	14		msb .. lsb												
RxOriginate	0-1	1	direct		X													
RxTurboScr (Note 7~~PError! Bookmark not defined.~~:V3.10)	0-1	1	direct															X
~~* BxAsyncESCInh_C (Rd Only, Note 7PError! Bookmark not defined.:V1.77)~~	0-1	1	direct														X	
RxIDRESCDat	**0-2**	2	direct											msb	lsb			
RxScr	**0-2**	2	direct									msb	lsb					
RxIDRCust	0-4	3	direct	16				msb .. lsb										
RxStatClkSource (Note 5~~PError! Bookmark not defined.~~, Note 7:V1.50)	0-1	1	direct				X											
RxStatClk10M (Note 5~~PError! Bookmark not defined.~~, Note 7:V1.50)	0-1	1	direct			X												
RxClosedNetPlusESC (Note 4~~PError! Bookmark not defined.~~, Note 7:V1.50)	0-1	1	direct		X													
RxPartialInsert	0-1	1	direct									X						
RxStatClk	0 and 1000 to 10000	17	4¼ Digit BCD										¼ d 4	msb .. digit 3 .. lsb				
				18	msb .. digit 2 .. lsb				msb .. digit 1 .. lsb					msb .. digit 0 .. lsb				
~~{Config memory number} (Note 6PError! Bookmark not defined.)~~	0-15	4	direct binary	20	msb .. Mem-N .. lsb													
RxPartialTS(0-31)	32 x (0-1)	32	1 bit per element (e)						e	e	e	e	e	e	e	e		

										31	30	29	28	27	26	25	24
				22	e23	e22	e21	e20	e19	e18	e17	e16	e15	e14	e13	e12
				24	e11	e10	e9	e8	e7	e6	e5	e4	e3	e2	e1	e0
~~RxIBSESCOh~~	0-255	8	direct	26	msb .. lsb											
RxTerDataRate	2400-4,920,000	28	7 Digit BCD										msb .. digit 6 .. lsb			
				28	msb .. digit 5 .. lsb				msb .. digit 4 .. lsb				msb .. digit 3 .. lsb			
				30	msb .. digit 2 .. lsb				msb .. digit 1 .. lsb				msb .. digit 0 .. lsb			
RxTS(0-31)	32 x (0-32)	192	6 bits per element 0-31 for specified elements 32 for unspecified elements (currently value of -1)	32	msb .. element 0 .. lsb						msb .. element 1 .. lsb					
				34	msb .. element 2 .. lsb						msb .. element 3 .. lsb					
				36	msb .. element 4 .. lsb						msb .. element 5 .. lsb					
				38	msb .. element 6 .. lsb						msb .. element 7 .. lsb					
				40	msb .. element 8 .. lsb						msb .. element 9 .. lsb					
				42	msb .. element 10 .. lsb						msb .. element 11 .. lsb					
				44	msb .. element 12 .. lsb						msb .. element 13 .. lsb					
				46	msb .. element 14 .. lsb						msb .. element 15 .. lsb					
				48	msb .. element 16 .. lsb						msb .. element 17 .. lsb					
				50	msb .. element 18 .. lsb						msb .. element 19 .. lsb					
				52	msb .. element 20 .. lsb						msb .. element 21 .. lsb					
				54	msb .. element 22 .. lsb						msb .. element 23 .. lsb					

				56	msb .. element 24 .. lsb						msb .. element 25 .. lsb					
				58	msb .. element 26 .. lsb						msb .. element 27 .. lsb					
				60	msb .. element 28 .. lsb						msb .. element 29 .. lsb					
				62	msb .. element 30 .. lsb						msb .. element 31 .. lsb					
(Spare)		36		64	--	--	--	--	--	--	--	--	--	--	--	--
				66	--	--	--	--	--	--	--	--	--	--	--	--
				68	--	--	--	--	--	--	--	--	--	--	--	--
TOTAL		**420**		**70**												

MSG: RxModRd & RxModWr																			
RxRSK	0-255, note 2_P	8	direct	0	msb .. lsb														
RxIFFreq	52-88MHz and 104-176MHz with 100Hz resolution	28	nnnnnn.n (KHz), 7 digit BCD														msb .. digit 6 .. lsb		
				2	msb .. digit 5 .. lsb						msb .. digit 4 .. lsb						msb .. digit 3 .. lsb		
				4	msb .. digit 2 .. lsb						msb .. digit 1 .. lsb						msb .. digit 0 .. lsb		
RxRSN	0-255, note 2_P	8	direct	6					msb .. lsb										
RxFECMode[2-0] bits[1-0]	**0-7**	2	direct bits[1-0]				msb	lsb											
RxFECRate[4-0] bits[1-0]	**0-31**	2	direct bits[1-0]		msb	lsb													
RxIntDepth	0-1	1	direct	8															X
RxSpectInv	0-1	1	direct															X	
RxMod[2-0] bits[1-0]	**0-7**	2	direct bits[1-0]												msb	lsb			
RxRSEn	**0-2**, note 2_P	2	direct								msb	lsb							
RxSweepWidth	**0-32**	6	direct		msb .. lsb														
* RxEqTx	0-1	1	direct	10						X									
~~{Config memory number} (Note 6~~ $_{P}$ ~~Error! Bookmark~~	0-15	4	direct binary														msb .. Mem-N .. lsb		

~~not defined.~~)																	
RxFECMode[2-0] bit[2] (Note 7PError! Bookmark not defined.:V2.87)	0-7	1	direct bit[2]						X								
RxMod[2-0] bits[2] (Note 7PError! Bookmark not defined.:V2.87)	0-7	1	direct bit[2]					X									
RxFECRate[4-0] bits[4-2] (Note 7PError! Bookmark not defined.:V2.87)	0-31	3	direct bits[4-2]		bit[4] .. bit[2]												
(Spare)		2									--	--					
(Spare)		12		12	--	--	--	--	--	--	--	--	--	--	--	--	--
TOTAL		**84**		**14**													

MSG: TxBBRd & TxBBWr

TxESCLvlCh1	+7 to -18 dBm (**0-250**)	8	(Lvl x 10) + 180	0	msb .. lsb
TxTerDataRate	2400-4,920,000	28	7 digit BCD		msb .. digit 6 .. lsb
				2	msb .. digit 5 .. lsb / msb .. digit 4 .. lsb / msb .. digit 3 .. lsb
				4	msb .. digit 2 .. lsb / msb .. digit 1 .. lsb / msb .. digit 0 .. lsb
TxESCLvlCh2	+7 to -18 dBm (**0-250**)	8	(Lvl x 10) + 180	6	msb .. lsb
TxAsyncBaud	0-15	4	direct (index to Baud array, see appendix D8)		msb .. lsb
TxIBSChanID	0-255	8	direct	8	msb .. lsb
TxAsyncChar	0-7	3	direct		msb .. lsb
TxDropIdle	0-1	1	direct		X
TxIBSESCOh	0-255	8	direct	10	msb .. lsb
TxClkMode	0-2	2	direct		msb .. lsb
TxDMXErrMon	0-3	2	direct		msb .. lsb
TxIBSSpareID	0-255	8	direct	12	msb .. lsb
TxIBSAuxDat	**0-2**	2	direct		msb .. lsb
TxIBSESCDat	0-3	2	direct		msb .. lsb

					b	b										
TxIBSStatID	0-255	8	direct	14					msb .. lsb							
TxIDRAuxDat	0-3	2	direct				msb	lsb								
TxClosedNetPlusESC (Note 4 PError! Bookmark not defined., Note 7:V1.50)	0-1	1	direct			X										
~~* BxAsyncESCInh_C (Rd only, Note 7 PError! Bookmark not defined.:V1.77)~~	0-1	1	direct		X											
TxBBMode	**0-23**	5	direct	16								msb .. lsb				
TxCustScr	0-31	5	direct				msb .. lsb									
TxScr	**0-2**	2	direct		msb	lsb										
TxFrmMode	0-7	3	direct	18										msb .. lsb		
TxIBSCust	0-7	3	direct								msb .. lsb					
TxIDRESCDat	0-2	2	direct						msb	lsb						
~~{Config memory number} (Note 6 PError! Bookmark not defined.)~~	0-15	4	direct binary		msb .. Mem-N .. lsb											
TxIDRCust	0-4	3	direct	20	msb .. lsb											
TxTurboScr (Note 7 PError! Bookmark not defined.:V3.10)	0-1	1	direct								X					
(Spare)		3						--	--	--						
~~TxIBSAuxOh~~	0-31	5	direct									msb .. lsb				
TxTS(0-31)	32 x (0-32)	192	6 bits per element 0-31 for specified elements	22	msb .. element 0 .. lsb						msb .. element 1 .. lsb					

			32 for unspecified elements (currently value of -1)														
				24	msb .. element 2 .. lsb						msb .. element 3 .. lsb						
				26	msb .. element 4 .. lsb						msb .. element 5 .. lsb						
				28	msb .. element 6 .. lsb						msb .. element 7 .. lsb						
				30	msb .. element 8 .. lsb						msb .. element 9 .. lsb						
				32	msb .. element 10 .. lsb						msb .. element 11 .. lsb						
				34	msb .. element 12 .. lsb						msb .. element 13 .. lsb						
				36	msb .. element 14 .. lsb						msb .. element 15 .. lsb						
				38	msb .. element 16 .. lsb						msb .. element 17 .. lsb						
				40	msb .. element 18 .. lsb						msb .. element 19 .. lsb						
				42	msb .. element 20 .. lsb						msb .. element 21 .. lsb						
				44	msb .. element 22 .. lsb						msb .. element 23 .. lsb						
				46	msb .. element 24 .. lsb						msb .. element 25 .. lsb						
				48	msb .. element 26 .. lsb						msb .. element 27 .. lsb						
				50	msb .. element 28 .. lsb						msb .. element 29 .. lsb						
				52	msb .. element 30 .. lsb						msb .. element 31 .. lsb						
(Spare)		24		54	--	--	--	--	--	--	--	--	--	--	--	--	
				56	--	--	--	--	--	--	--	--	--	--	--	--	
TOTAL		**348**		**58**													

MSG: TxModRd & TxModWr																
TxRSK	0-255, note 2_P	8	direct	0	msb .. lsb											
TxIFFreq	52-88MHz and 104-176MHz with 100Hz resolution	28	nnnnnn.n (KHz), 7 Char BCD										msb .. digit 6 .. lsb			
				2	msb .. digit 5 .. lsb				msb .. digit 4 .. lsb				msb .. digit 3 .. lsb			
				4	msb .. digit 2 .. lsb				msb .. digit 1 .. lsb				msb .. digit 0 .. lsb			
TxRSN	0-255, note 2_P	8	direct	6					msb .. lsb							
TxFECMode[2-0] bits[1-0]	**0-7**	2	direct bits[1-0]				msb	lsb								
TxFECRate[4-0] bits[1-0]	**0-31**	2	direct bits[1-0]		msb	lsb										
* TxPower	**0-250**	8	Lvl x -10	8					msb .. lsb							
* TxCarrier	0-3	2	direct				msb	lsb								
(Spare)		2			--	--										
TxIntDepth	0-1	1	direct	10						X						
TxMod[2-0] bits[1-0]	**0-7**	2	direct bits[1-0]					msb	lsb							
TxRSEn	**0-2**, note 2_P	2	direct			msb	lsb									

						b											
TxSpectInv	0-1	1	direct		X												
{Config memory number} (Note 6 PError! Bookmark not defined.)	0-15	4	direct binary										msb .. Mem-N .. lsb				
TxFECMode[2-0] bit[2] (Note 7 PError! Bookmark not defined.:V2.87)	0-7	1	direct bit[2]									X					
TxMod[2-0] bit[2] (Note 7 PError! Bookmark not defined.:V2.87)	0-7	1	direct bit[2]								X						
TxFECRate[4-0] bits[4-2] (Note 7 PError! Bookmark not defined.:V2.87)	0-31	3	direct bits[4-2]	12										bit[4] .. bit[2]			
(Spare)		9			--	--	--	--	--	--	--	--	--				
TOTAL		**84**		**14**													

MSG: UnitCfgRd & UnitCfgWr																
~~BxAutoLogFlags~~	0-1023	8	direct (bits 0-7)	0	(bit 7) msb .. lsb (bit 0)											
~~BxAutoLogPeriod~~	0-9999	16	4 digit BCD										msb .. digit 3 .. lsb			
				2	msb .. digit 2 .. lsb				msb .. digit 1 .. lsb				msb .. digit 0 .. lsb			
BxTerrIntRead_U (UnitCfgRd) or BxTerrIntWrite (UnitCfgWr)	0-255	8	direct	4	msb .. lsb											
~~BxAutoLogFlags (Note 7~~ ~~P~~~~Error! Bookmark not defined.~~:V2.09)	0-1023	2	direct (bits 8&9)										bit 9	bit 8		
BxMaxMFPeriod	**0-15000**	14	direct												b13	b12
				6	bit 11 .. bit 0											
RxEbNoWarn	0.0 to 25.5 (0-255)	8	Value x 10	8	msb .. lsb											
RxIMXErrMon	0-3	2	direct										msb	lsb		
RxMinSlipHrs	0-9999	14	direct												b13	b12
				10	bit 11 .. bit 0											
* ID_VirtInt	0-255	8	direct	12					msb .. lsb							
BxUnitSetup	0-1	1	direct					X								
BxESCIntfc	**0-4**	3	direct		msb .. lsb											
BxBERWarn	Note 1P	12	Note 1	14	msb .BIN exp. lsb				msb .BCD mant 1. lsb				msb .BCD mant 0. lsb			
BxTerrIntMode	0-31	5	direct	16									msb .. lsb			

~~BxFaultMode~~	0-15	4	direct						msb .. lsb													
BxIBSMultiBA	**0-4**	3	direct				msb .. lsb															
~~BxRelayMode~~	0-15	4	direct	18															msb .. lsb			
RxSigBlockCode	0-3	2	direct													msb	lsb					
~~RxAGCOPScale~~	0-15	4	direct					msb .. lsb														
~~TxBackAlmMode~~	0-3	2	direct		msb	lsb																
~~BxAISAction~~	0-1	1	direct	20																		X
~~BxAuxIntfc~~	0-1	1	direct																	X		
~~BxBERAction~~	0-1	1	direct																X			
~~BxG732CRCAcq~~	0-1	1	direct															X				
~~BxG732EBits~~	0-1	1	direct														X					
~~BxG732TerrCRC~~	0-1	1	direct													X						
~~BxIFImpedance~~	0-1	1	direct								X											
~~BxOpDisp~~	0-1	1	direct							X												
~~BxTRSpoof~~	0-1	1	direct						X													
~~BxTS0G.732Spares~~	0-1	1	direct					X														
~~BxUpgradeMsg~~	0-1	1	direct				X															
~~BxX50FullAIS~~	0-1	1	direct		X																	
~~FanMode~~	0-1	1	direct	22							X											
~~RxBuffAutoCent~~	0-1	1	direct							X												
~~RxBuffMFSlip~~	0-1	1	direct						X													
~~RxBERMon~~	0-4	3	direct				msb .. lsb															
{Config memory number} (Note 6~~PError! Bookmark not defined.~~)	0-15	4	direct binary																msb .. Mem-N .. lsb			

(Spare)		2										--	--				
~~Bx10MHzRefOffset (Note 7~~ ~~Error! Bookmark not defined.~~ ~~:V2.87)~~	0-100	8	Value+128. Incoming values of 0-127 and 229-255 are allowed but ignored to preserve compatibility with earlier drivers not supporting this variable.	24					msb .. lsb								
(Spare)		4			--	--	--	--									
TOTAL		**156**		**26**													

MSG: UnitInfoRd

~~{ID_AuxAud / ID_Q308}~~[A]	0-7	3	direct	0														msb .. lsb	
~~ID_Dig[2-0]~~	0-7	3	direct												msb .. lsb				
ID_Intfc	0-63	6	direct		msb .. lsb														
* ID_VirtInt	0-255	8	direct	2					msb .. lsb										
~~RxBearer_H~~	0-1	1	direct					X											
~~ID_IFU~~	0-3	2	direct			msb	lsb												
~~TxIntClk_H~~	0-1	1	direct		X														
(Spare)		4		4	--	--	--	--											
SerialNumber	0-99,999	20	5 digit BCD						msb .. digit 4 .. lsb				msb .. digit 3 .. lsb						
				6	msb .. digit 2 .. lsb				msb .. digit 1 .. lsb				msb .. digit 0 .. lsb						
* {ModelNumber} (fromSAFModel[1-0] & SAFManf[2-0]) upper case, right justified packed with spaces	4 chars	24	4 digit (ASCII-48) giving (0-63)	8	msb .. left char .. lsb								msb .. 2nd left char .. lsb						
				10	msb .. 2nd right char .. lsb								msb .. Right char .. lsb						
* SoftwareVersion	0.00 to 9.99	12	3 digit BCD	12	msb .. digit 2 .. lsb				msb .. digit 1 .. lsb				msb .. digit 0 .. lsb						
~~{ID_RF[1-0] / ID_Q311}~~[A]	0-3	2	direct	14													msb	lsb	
~~{ID_RF[4-2] / ID_Q313}~~[A]	0-7	3	direct											msb .. lsb					
~~{ID_RF[6-5] / ID_Q314}~~[A]	0-3	2	direct							msb	lsb								
~~{ID_RF[7] / TxHiCurrentSupply}~~[A]	0-1	1	direct						X										
~~{ID_RF[11-8] / Spare}~~[A]	0-15	4	direct		msb .. lsb														

~~{ID_RF[15-12] (Note 7~~[P]~~Error! Bookmark not defined.:V1.56) / Spare}~~[A]	0-15	4	direct	16			msb .. lsb										
~~ID_Q303 (Note 5~~[P]~~Error! Bookmark not defined., Note 7:V1.56)~~	0-1	1	direct			X											
~~ID_Q304 (Note 5~~[P]~~Error! Bookmark not defined., Note 7:V1.56)~~	0-1	1	direct		X												
~~ID_IDR (Note 5~~[P]~~Error! Bookmark not defined., Note 7:V1.56)~~	0-7	3	direct												msb .. lsb		
~~ID_Dig[4-3] (Note 7~~[P]~~Error! Bookmark not defined.:V2.87)~~	0-3	2	direct									b it 4	b it 3				
~~{Format indicator 0/1}Note **A**~~[below] ~~(Note 7~~[P]~~Error! Bookmark not defined.:V3.00)~~	0-1	1	direct							X							
TOTAL		**108**		**18**													

Note **A**: See note 7[P]:Message response in one of two formats (corresponding to P400 / early P300, or later P300 / P310) to reflect the different subassemblies in each product. The single bit "Format Indicator" indicates which of the two formats is returned (so the information may be interpreted correctly). The format shown in the above table shows {"description for format=0" / "description for format=1"}

MSG: UnitStatRd

BxConfigError	0-255	8	direct	0					msb .. lsb							
TxTrafFault_U	0-15	4	direct		msb .. lsb											
~~BxConfigInfo~~	0-255	8	direct	2					msb .. lsb							
TxTrafWarn_U	0-15	4	direct		msb .. lsb											
BxConfigWarn	0-255	8	direct	4					msb .. lsb							
BxUnitFault_U	0-15	4	direct		msb .. lsb											
(Spare)		1		6	--											
{TxRunTime} (calculated from TxlastBreak)	0.0-99 secs to days	11	UEnn: U=units: 0=secs, 1=mins, 2=hrs, 3=days E=1 bit exponent: 0=n.n, 1=nn nn 2 digit BCD of `OK` time			msb.U.lsb		E	msb . mant 1 . lsb				msb . mant 0 . lsb			
{RxRunTime} (calculated from RxlastBreak)	0.0-99 secs to days	11		8		msb.U.lsb		E	msb . mant 1 . lsb				msb . mant 0 . lsb			
(Spare)		1			--											
RxTrafFault_U	0-31	5	direct	10								msb .. lsb				
BxUnitWarn_U	0-7	3	direct						msb .. lsb							
(Spare)		4			--	--	--	--								
RxEbNo_U	0.0 to 25.5 (0-255)	8	Value x 10	12	msb .. lsb											
{RxEbNo_U Prefix / validation}	0-3	2	0→"=", 1→"<", 2→">", 3→"invalid" When "invalid" RxEbNo_U gives 0=unavailable, 1=dem unlocked, 2=FEC unlocked, 3=lock det timeout)										msb	lsb		
RxLastBreak	MDHMS	26	MDHMS												M	M

			Month=4 bits, Date=5 Hour=5, Mins=6, Secs=6													3	2
				14	M 1	M 0	msb .. Date ..lsb						msb .. Hrs ..lsb					
				16	msb .. Mins .. lsb								msb .. Secs ..lsb					
RxBuffFill_U	0-100	7	direct	18	msb .. lsb													
~~RxBERSrc_C~~	**0-4**	3	direct										msb .. lsb					
TxLastBreak	MDHMS	26	MDHMS Month=4 bits, Day=5 Hour=5, Mins=6, Secs=6													M 3	M 2
				20	M 1	M 0	msb .. Date ..lsb						msb .. Hrs ..lsb					
				22	msb .. Mins .. lsb								msb .. Secs ..lsb					
BxRelayStatus_U	0-15	4	direct	24	msb .. lsb													
~~RxBuffOver_U~~	0-99	7	direct						msb .. lsb									
RxFreqOff_U	±99.9KHz	13	snn.n s=1 bit sign 0=+ve, 1=-ve nn.n= 3 digit BCD mag															s
				26	msb .. digit 3 .. lsb				msb .. digit 2 .. lsb						msb .. digit 0 .. lsb			
RxUserErrRate_U	Note 1_P	12	Note 1	28	msb .BIN exp. lsb				msb .BCD mant 1. lsb						msb .BCD mant 0. lsb			
TxCarrierStat_U	0-7	3	direct	30												msb .. lsb		
~~RxBuffUnder_U~~	0-99	7	direct				msb .. lsb											
Bx1FOR1Standby_U	0-1	1	direct			X												
{Device configuring / busy}	0-1	1	0=Normal, 1=busy/reconfiguring		X													
RxUserErrRate_U	Note 1_P	2	Note 1	32							S 1	S 0						
(Spare)		1			--													

RxTrafWarn_U	0-31	5	direct		msb .. lsb
~~Alog_U(1-8) (0-255)~~ ~~Element 8 = +5V digital~~ ~~Element 7 = -5V~~ ~~Element 6 = -12V~~ ~~Element 5 = +12V~~ ~~Element 4 = +5V analog (P300 n/a)~~ ~~Element 3 = AGC~~ ~~Element 2 = Temperature~~ ~~Element 1 = Confidence check~~	8 x 0-255	64	each value `direct` except element 2 encoded as follows (Bits[7..0]): Bits[7..6]: other bits contain: 0=Actual value, 1=<value, 2=>value, 3=Error code. Bits[5..0]: When `value`: - 1x(Level+21)dBm ie 0..63 represent -21 to -84dBm When `error`: 0=Unavail, 1=Demod not locked, 2=FEC not locked, 3=lock detect timeout.		b7 .. element 8 .. b4
				34	b3 .. element 8 .. b0 / msb .. element 7 .. lsb
				36	msb .. element 6 .. lsb / b7 .. element 5 .. b4
				38	b3 .. element 5 .. b0 / msb .. element 4 .. lsb
				40	msb .. element 3 .. lsb / b7 .. element 2 .. b4
				42	b3 .. element 2 .. b0 / msb .. element 1 .. lsb
BxUnitTemperature_U Note 7[PError! Bookmark not defined.]:V2.87	-55 to +200	8	(Temp in C) + 55	44	msb .. lsb
(Spare)		4			-- -- -- --
TOTAL		**276**		**46**	

MSG: UnitTestRd & UnitTestWr

BxLoopMode	0-7	3	direct	0										msb .. lsb				
BxPRBSMode	0-3	2	direct									msb	lsb					
BxPRBSPattern	0-3	2	direct						msb	lsb								
BxTestMode	**0-3**	3	direct			msb .. lsb												
~~RxPRBSSingleShot~~	0-1	1	direct		X													
(Spare)		1		--														
(Spare)		2		2				--	--									
~~BxEIA530Loop_U (Rd only)~~	0-3	2	direct			msb	lsb											
(Spare)		2		--	--													
TOTAL		**18**		**3**														

P4xx High Level Variable {Derived Value / action}	Range	# bits	Encoding in message	Body Character	+0						+1					
				Bits:	5	4	3	2	1	0	5	4	3	2	1	0
MSG: ActionWr																
{Buffer centre}	0-1	1	1=Action	0						X						
{PRBS Inject Error} (Note 7 PError! Bookmark not defined.:V2.34)	0-1	1	1=Inject						X							
{Test Stop/Start/Restart} (Note 7 PError! Bookmark not defined.:V2.34)	0-3	2	0=None, 1=Stop, 2=Start, 3=Restart				msb	lsb								
{Zero slip counters}	0-1	1	1=Action			X										
{Reset Unit}	0-1	1	1=Action		X											
{Accept current config warning}	0-1	1	1=Accept													X
{Accept all config warnings}	0-1	1	1=Accept												X	
{Accept current config info}	0-1	1	1=Accept											X		
{Accept all config info}	0-1	1	1=Accept										X			
(Spare)		1										--				
(Spare)	0-1	1									--					
~~{Request Setup check}~~	0-1	1	1=Request	2						X						
~~{Force `Setup flag`}~~	0-1	1	0=No action (ie NOT force low) 1=Force BXUNITSETUP=1						X							
~~{Store config to memory N}~~	0-1	1	1=Store					X								
~~{Recall memory N}~~	0-1	1	1=Recall				X									
{1:1 Give Away}	0-1	1	1=Give 1:1 control			X										
{Start SAF `5 day feature test} (Note 7 PError! Bookmark not defined.:nya)	0-1	1	1=Start		X											

P4xx High Level Variable {Derived Value / action}	Range	# bits	Encoding in message	Body Character	+0						+1					
				Bits:	5	4	3	2	1	0	5	4	3	2	1	0
(spare)		2									--	--				
{Memory Number}	0-15	4	0-9 for memories 1-10										msb .. Mem-N .. lsb			
TOTAL		**24**		**4**												

MSG: TxPwrRd & TxPwrWr																		
* TxPower	**0-250**	8	Lvl x -10	0					msb .. lsb									
* TxCarrier	0-3	2	direct				msb	lsb										
{Ignore TxPower} (Note 7PError! Bookmark not defined.:V1.77)	0-1	1	0=Normal, 1=Ignore TxPower (so becomes just carrier on /off)			X												
(Spare)		1			--													
TOTAL		**12**		**2**														

MSG: ShortDeviceRd																		
* {ModelNumber} (fromSAFModel[1-0] & SAFManf[2-0]) upper case, right justified packed with spaces	4 chars "Ch3 Ch2 Ch1 Ch0" eg "P440"	24	4 digit (ASCII-48) giving (0-63)	0	msb .. Char 3 .. lsb						msb .. Char 2 .. lsb							
				2	msb .. Char 1 .. lsb						msb .. Char 0 .. lsb							
* {SoftwareVersion}	0.0 to 9.9	8	2 digit (ASCII-48) giving (0-63)	4	msb .. units .. lsb						msb .. tenths .. lsb							
TOTAL		**36**		**6**														

MSG: TxLBandRd/Wr (see Note 7 PError! Bookmark not defined.:V2.87 re whole Tx/RxLBandRd/Wr messages)																
TxDCVoltage	0-1	1	direct	0				X								
Tx10MHzRef	0-1	1	direct			X										
(spare)		2			--		--									
TxIFFreq	from 50MHz to 2150MHz with 100Hz resolution	32	nnnn.nnnn (MHz), 8 Char BCD		msb .. digit 7 .. lsb								msb .. digit 6 .. lsb			
				2	msb .. digit 5 .. lsb				msb .. digit 4 .. lsb				msb .. digit 3 .. lsb			
				4	msb .. digit 2 .. lsb				msb .. digit 1 .. lsb				msb .. digit 0 .. lsb			
~~TxSHFPower~~	99.9 to -99.9	1	Sign bit, 0=+ve, 1=-ve	6								X				
~~TxSHFPowerUnits~~	0-1	1	0="dBM", 1="dBW"								X					
~~TxSHFPowerRadiated~~	0-1	1	0="TxPwr", 1="EIRP"							X						
(spare)		5			--	--	--	--	--							
~~TxSHFOffset~~	0-65535	16	direct										bit 15 .. bit 12			
				8	bit 11 .. (lower order bits) .. bit 0											
~~TxSHFPower~~	99.9 to -99.9	12	3 digit BCD	10	msb .. tens .. lsb				msb .. units .. lsb				msb .. tenths .. lsb			
TxDCCurrentUserMax	0-4000	12	direct	12	msb .. lsb											
TxDCCurrentUserMin	0-4000	12	direct	14	msb .. lsb											
(spare)		7		16					--	--	--	--	--	--	--	
~~{Config memory number} (Note 6 PError! Bookmark not defined.)~~	0-15	4	direct binary		msb .. Mem-N .. lsb											
TxDCCurrent_U (Read Only)	0-6250	13	direct													bit 12
				18	bit 11 .. bit 0											

~~TxRFPower_U (Read only)~~	00.0 to -35.0	12	3 digit BCD (implied negative)	20	msb .. tens .. lsb				msb .. units .. lsb				msb .. tenths .. lsb										
TOTAL		**132**		**22**																			

MSG: RxLBandRd/Wr																	
RxDCVoltage	0-2	2	direct	0			msb	lsb									
Rx10MHzRef	0-1	1	direct			X											
~~RxDCCurrentSense~~	0-1	1	direct		X												
RxIFFreq	from 50MHz to 2150MHz with 100Hz resolution	32	nnnn.nnnn (MHz), 8 Char BCD						msb .. digit 7 .. lsb				msb .. digit 6 .. lsb				
				2	msb .. digit 5 .. lsb				msb .. digit 4 .. lsb				msb .. digit 3 .. lsb				
				4	msb .. digit 2 .. lsb				msb .. digit 1 .. lsb				msb .. digit 0 .. lsb				
(spare)		8		6	--	--	--	--	--	--	--	--					
~~RxSHFOffset~~	0-65535	16	direct binary										bit 15 .. bit 12				
				8	bit 11 .. (lower order bits) .. bit 0												
(spare)		8		10					--	--	--	--	--	--	--	--	
~~{Config memory number) (Note 6~~ ~~Error! Bookmark not defined.~~ ~~)~~	0-15	4	direct binary		msb .. Mem-N .. lsb												
TOTAL		**72**		**12**													

MSG: UnitResRd																			
~~RxDFRErrRateIn_U~~	Note 1_P	12	Note 1	0	msb .BIN exp. lsb								msb .BCD mant 1. lsb				msb .BCD mant 0. lsb		
~~RxFECErrRateIn_U~~	Note 1_P	12	Note 1	2	msb .BIN exp. lsb								msb .BCD mant 1. lsb				msb .BCD mant 0. lsb		
~~RxFECErrRateOut_U~~	Note 1_P	12	Note 1	4	msb .BIN exp. lsb								msb .BCD mant 1. lsb				msb .BCD mant 0. lsb		
RxIBSChanID_U	0-255	8	direct	6					msb .. lsb										
~~RxIMXErrSrc_C~~	0-3	2	direct				msb	lsb											
~~TxDMXErrSrc_C~~	0-3	2	direct		msb	lsb													
~~RxDFRErrRateIn_U~~	Note 1_P	2	Note 1	8	S1	S0													
~~RxFECErrRateIn_U~~	Note 1_P	2	Note 1				S1	S0											
RxIBSSpareID_U	0-255	8	direct						msb .. lsb										
~~RxFECErrRateOut_U~~	Note 1_P	2	Note 1	10	S1	S0													
~~RxIMXErrRate_U~~	Note 1_P	2	Note 1				S1	S0											
RxIBSStatID_U	0-255	8	direct						msb .. lsb										
~~RxIMXErrRate_U~~	Note 1_P	12	Note 1	12	msb .BIN exp. lsb								msb .BCD mant 1. lsb				msb .BCD mant 0. lsb		
~~RxRSErrRateIn_U~~	Note 1_P	12	Note 1	14	msb .BIN exp. lsb								msb .BCD mant 1. lsb				msb .BCD mant 0. lsb		

~~RxRSErrRateOut_U~~	Note 1_P	12	Note 1	16	msb .BIN exp. lsb				msb .BCD mant 1. lsb				msb .BCD mant 0. lsb				
~~TxDMXErrRate_U~~	Note 1_P	12	Note 1	18	msb .BIN exp. lsb				msb .BCD mant 1. lsb				msb .BCD mant 0. lsb				
~~RxRSErrRateIn_U~~	Note 1_P	2	Note 1	20	S 1	S 0											
~~RxRSErrRateOut_U~~	Note 1_P	2	Note 1				S 1	S 0									
~~TxDMXErrRate_U~~	Note 1_P	2	Note 1						S 1	S 0							
(Spare)		6									--	--	--	--	--	--	
TOTAL		**132**		**22**													

Chapter 5 Management Information Base

The modem uses two SNMP Management Information Bases (MIBs). The Paradise MIB is common to all Paradise Datacom equipment and defines the top-level object identifiers for each piece of equipment (such as modems, SSPAs, etc.). The modem MIB defines the object identifiers that are specific to the modem M&C controls. The SNMP Management Information Base (MIB) can be downloaded directly from the modem (via the 'Download MIB files' hyperlink at the foot of the *Edit->Unit->M&C->SNMP* web page) or can be obtained from Customer Support.

The structure of the MIB follows that of the web user interface menus. This allows a developer using SNMP to create an alternative user interface to relate OIDs to specific Paradise user controls. Note that since OIDs can appear once only in the MIB, whereas the equivalent controls may appear on more than one web page, some OIDs may appear to be missing when comparing the web user interface menus with the MIB structure. (This problem is mainly restricted to the controls on the Status and View web pages.) A partial view of the MIB structure is shown in **Figure 5-1**.

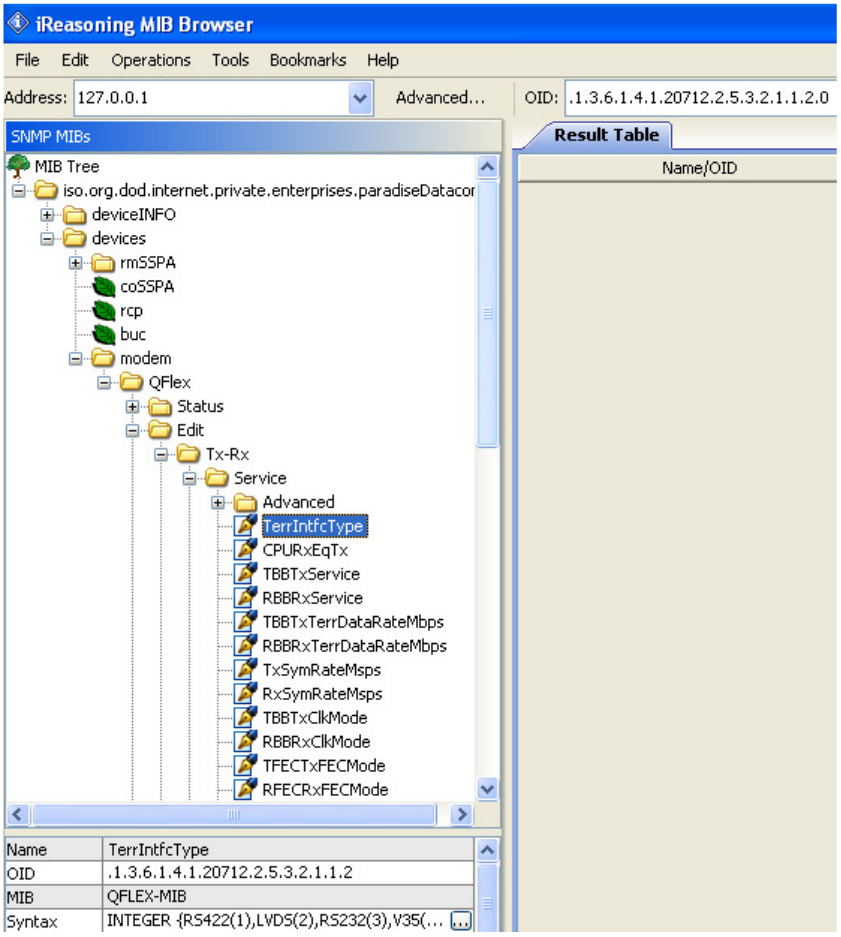


Figure 5-1 Q-Series SNMP MIB Structure (shown in MIB Broswer)