Interface Control Document Part 1: Software Protocol Specification

TXMission Ltd.

September 2019

Contents

Software Protocol Specification

Introduction

This interface control document (the first of two) defines a software protocol for communicating with our **Quest™** onboard and **Connect™** ground station modems, shown in **Figure 1**. A second interface control document (Part 2) defines the hardware interfaces (connector pinouts and electrical and mechanical characteristics).

Figure 1: Quest™ Onboard Smallsat Modem and Connect™ Ground Station Modem

A modem system block diagram is shown in **Figure 2**.

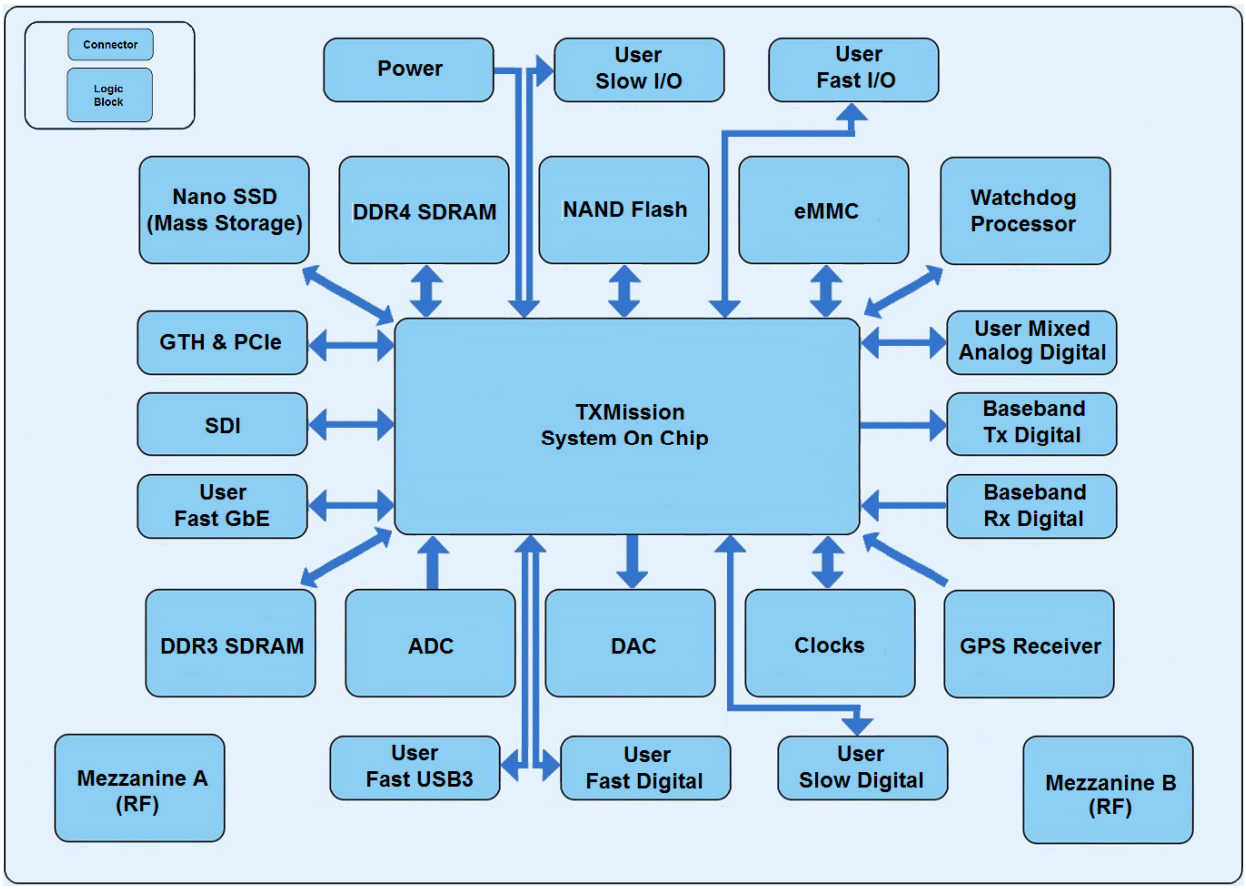

Figure 2: Modem System Block Diagram

The One and Only Protocol for Smallsats! (OOPS!) is an open-source protocol for communicating with low and medium-earth satellites (to which, for convenience, we have assigned the rather arbitrary collective title of 'smallsats'). OOPS! provides an alternative to using the built-in web user interfaces provided by the modems. OOPS! was created by TXMission to address the notable shortcomings found in existing space protocols with respect to their ability to be applied to the unique requirements of the New Space industry, which continues to undergo an unprecedented revolution in technology. Following its initial development, OOPS! is now being freely offered to the space community as an open standard. It is anticipated that it will continue to be developed in future under the auspices of an open industry forum that TXMission is committed to helping to set up and support.

The scope of OOPS! is to control and monitor all onboard subsystems and the ground-segment equipment counterparts that are required to implement an end-to-end smallsat communications system. It therefore covers both uplink and downlink, housekeeping data, telemetry data and other requirements. It is suitable for a wide range of applications, including earth observation, telecoms, IOT and 5G.

OOPS! operates at the application layer with respect to both the TCP/IP and OSI models. It is compatible with various transport mechanisms including TCP/IP, CCSDS packet telemetry (specified in CCSDS 102.0-B-4) and CubeSat Space Protocol (CSP) (which is a lightweight alternative to TCP/IP). Internally, TXMission uses OOPS! in conjunction with automated SSH control via TCP/IP as part of our MissionSpan network management system (we take the view that TCP/IP is set to become the dominant smallsat communications technology). We use CCSDS as an optional complementary technology that provides a mechanism for transporting OOPS! and other data from specific onboard sources to specific handler processes on the ground. OOPS! leaves all security and reliability issues to the other layers.

Much of the value of OOPS! lies in the fact that it is a lightweight protocol that is intuitive to use (being human readable), easy to implement and readily extensible. From the outset it has supported the ability for any organisation to add its own command extensions with minimum formality. While an open forum is still in the process of being formed, TXMission is happy to collate and make public any new OOPS! documentation that is provided to us by third parties.

Overview

OOPS! is a command line protocol that uses the grammatical conventions enshrined in both the Linux and Windows command line languages and by many others in addition. Newcomers will therefore often find it provides a familiar framework, as its structure and usage tend to resonate with prior experiences of other command languages.

Although OOPS! is a command line protocol that lends itself to being typed in by a user via a telnet or SSH session (or through a serial interface) it can also be incorporated into fully automated operational and test management systems.

OOPS! currently recognises the following smallsat subsystems:

- eps: electrical power system
- radio: communications system (typically a combination of a satellite modem/Software Defined Radio and associated RF/antenna).
- adcs: attitude determination and control system
- obc: onboard computer (sometimes referred to as a flight computer)
- payload: payload system

The above terms are used as command prefixes to identify which subsystem a command relates to. For example, a smallsat can be made to generate a carrier by sending it the following command:

radio.set carrier on

The OOPS! command handler remembers the last subsystem addressed and the prefix is therefore not required on subsequent consecutive commands to the same subsystem. (The default prior to any prefix being used for the first time is to send the commands to the radio subsystem.) For example, if at a later point we want to switch the carrier off, we could send the abbreviated command:

set carrier off

If, however, we had addressed commands to other subsystems in the intervening period then we would have to issue the full form of the command for it to be recognised:

radio.set carrier off

Note that commands are always in lower case. Each command must be terminated with a new line even when control is automated. The definition of a new line varies with operating system, being a line feed (‘\n’) in Linux and a carriage return and line feed (‘\r\n’) in Windows.

Commands are executed at the time they are received unless *batch* mode is enabled:

batch on

In batch mode, all commands that are received are held pending by the relevant subsystem until an explicit *submit* command is received:

submit

Executing commands together as a batch can significantly reduce service downtime and prevent the potential for undesirable side effects that could result from partial reconfiguration. If batch mode is not used, it is recommended with respect to the radio that the transmit carrier is muted during reconfiguration of the modulator in order to prevent the potential for a carrier with undefined characteristics being generated.

Batch mode can be switched off as follows:

batch off

Every command will generate a response and the user should wait for a response before sending the next command. Responses end with a new line followed by '# ', which indicates that the equipment is ready to receive another command. If a command fails then the response will consist of a '!' followed by an error message (which is then terminated by the usual new line and '# '). An example of a command with its response is as follows:

 # get carrier

Off

#

This indicates that the modulator's carrier is currently muted.

Configurations for specific subsystems can be stored (in non-volatile memory) for later use. The *touch* command is used to create a new configuration where each setting for the subsystem is set to its default value. To modify a stored configuration, the redirection operator '>' can be used with any regular command. For example:

touch myconfig

set service dvbs2 > myconfig

creates a configuration file called *myconfig* and then sets the service to DVB-S2. It is also possible to change the current active configuration via a series of commands and then issue a *save* command in order to store the current settings to a named configuration:

save mynewconfig

A stored configuration can be enabled as the current active configuration using the *select* command:

select myconfig

Configurations can be viewed and deleted as follows:

 # dir

myconfig1

myconfig2

myconfig3

 # del myconfig1

#

A list of all subsystem commands can be displayed by issuing the *help* command:

help

Finally, note that the radio subsystem has its own real-time clock, which includes a battery and maintains the time even when powered down. There is currently no OOPS! command to set this clock but it can be set by the user through operating system commands (please contact us for more details).

Radio Commands

This section specifies the commands that can be used with respect to the smallsat's radio subsystem and the ground system counterpart. Appendix A (TXMission Radio Performance) specifies limitations in relation to specific TXMission products (such as the maximum valid data rate). These limitations must be observed when entering commands.

batch

batch [off | on]

Switches batch command processing mode off or on. The default setting is off. If no parameter is supplied then a response is returned that indicates the current setting (i.e. whether batch mode is off or on).

Commands are executed at the time they are received unless batch mode is enabled. When batch mode is enabled then all commands that are received are held pending by the radio until a command is received that has a *submit* suffix (e.g. *set carrier on submit*) or an explicit *submit* command is executed.

Executing commands together as a batch can significantly reduce service downtime and prevent the potential for undesirable side effects that could result from partial reconfiguration. If batch mode is not used, it is recommended that in respect of the radio that the transmit carrier is muted during reconfiguration of the modulator in order to prevent the potential for a carrier with undefined characteristics being generated.

Example:

 # batch on

 # batch

On

#

del

del <filename>

Arguments:

<filename> Specifies the name for an existing radio configuration.

See related configuration commands: *touch*, *save*, *select* and *dir*, which allow multiple radio configurations to be created, manipulated, saved and recalled. Configurations are virtual files that are held in an embedded database on each radio.

Example:

 # dir

myconfig1

myconfig2

myconfig3

 # del myconfig1

#

get alarms

get alarms

Returns a list of the currently active radio alarms.

Example:

 # get alarms

Demodulator FEC loss of sync

Demodulator unlocked

#

get carrier

get carrier

Indicates the status of the radio's carrier (i.e. whether a carrier is being generated).

Example:

 # get carrier

On

#

get freq

get freq [tx] [rx]

tx	Returns the center frequency of the transmitted carrier.
rx	Returns the center frequency of the received carrier.
	If neither *tx* nor *rx* is specified then the command returns both frequencies (transmit coming before receive).

Indicates the current transmit and/or receive carrier frequency in MHz.

Example:

 # get freq

2325.000000 2225.000000

#

get interface

get interface [tx] [rx]

Arguments:

tx	Returns the physical interface selected for the transmit path of the radio.
rx	Returns the physical interface selected for the receive path of the radio.
	If neither *tx* nor *rx* is specified then the command returns both selected interfaces (transmit coming before receive).

Indicates the current physical interfaces associated with the transmission and reception of data in the radio.

Example:

 # get interface

eth lvds

#

get modcod

get modcod [tx] [rx] [--frame] [--pilots]

Options:

-f, --frame	Returns the DVB-S2/S2X frame size. This option is not used with other services.
-p, --pilots	Returns the DVB-S2/S2X pilots setting. This option is not used with other services.

Arguments:

tx	Returns the modcod associated with the transmit path of the radio.
rx	Returns the modcod associated with the receive path of the radio.
	If neither *tx* nor *rx* is specified then the command returns information for both transmit and receive paths of the radio (transmit coming before receive).

Indicates the current modcods associated with the transmit and/or receive paths in the radio.

For a list of the returned modcod values, see *set modcod*.

Examples:

 # get modcod tx

qpsk-1/2

 # get modcod rx -f -p

qpsk-1/2 short Off

 # get modcod -f -p

qpsk-1/2 short Off 8psk-3/4 normal On

#

get power

get power

Indicates the power level of the transmitted carrier in dBm.

Example:

 # get power

-22.3

#

get rate

get rate [txdata] [txsym] [rxdata] [rxsym]

Arguments:

txdata	Returns the transmit data rate in Mbps.
txsym	Returns the transmit symbol rate in Msps.
rxdata	Returns the receive data rate in Mbps.
rxsym	Returns the receive symbol rate in Msps.

Indicates the current transmit and receive data and symbol rates. If no parameters are provided then all data and symbol rates are returned in the order shown.

Examples:

 # get rate txdata txsym rxdata rxsym

10.000000 4.134780 2.000000 0.988343

 # get rate

10.000000 4.134780 2.000000 0.988343

 # get rate rxsym

5.000000

get rolloff

get rolloff [tx] [rx]

Arguments:

tx	Returns the carrier roll-off being used on the transmit path of the radio.
rx	Returns the carrier roll-off being used on the receive path of the radio.
	If neither *tx* nor *rx* is specified then the command returns both roll-off factors (transmit coming before receive).

Indicates the current carrier roll-off factors associated with the transmit and/or receive paths in the radio.

Example:

 # get rolloff

35% 25%

#

get service

get service [tx] [rx]

Arguments:

tx	Indicates the service being used on the transmit path of the radio.
rx	Indicates the service being used on the receive path of the radio.
	If neither *tx* nor *rx* is specified then the command will return both services (transmit coming before receive).

Indicates the current service being provided by the transmit and/or receive paths in the radio.

Example:

 # get service tx

dvbs2x

#

get status

get status [tx] [rx]

Arguments:

tx	Returns status information for the transmit path of the radio. In order, the information that is returned is: *Tx carrier status* (*On* or *Off*), *Tx output power* (dBm), *Tx data rate* (Mbps), *Tx symbol rate* (Msps), *Tx frequency* (MHz).
rx	Returns status information for the receive path of the radio. In order, the information that is returned is: *Rx Es/No* (dB), *Rx Eb/No* (dB), *Rx wanted power* (dBm), *Rx composite power* (dBm), *Rx data rate* (Mbps), *Rx symbol rate* (Msps), *Rx frequency* (MHz), *Rx frequency offset* (Hz), *Rx frame count*, *Rx frame errors*. If neither *tx* nor *rx* is specified, then the command will return status information for both services (transmit coming before receive).

Provides current status information for the transmit and/or receive paths in the radio.

Examples:

 # get status tx

On -30.9 64.123456 29.457723 2205.244302

 # get status rx

10.3 7.8 -37.6 -48.7 64.373321 25.448976 2200.066912 13.2 10134 2

 # get status

On -30.9 64.123456 29.457723 2205.244302 10.3 7.8 -37.6 -48.7 64.37 25.44 2200.066912 13.2 10134 2

#

help

help

This command displays a full list of the available commands pertaining to the radio subsystem.

reset

reset

This initiates a soft reset of the radio, which is similar in its operation to a hard reset (power cycle).

save

save <filename>

Arguments:

<filename> Specifies the name for a new or existing radio configuration. Valid names are as per Linux filenames. There is no concept of nested folders and all configurations exist in the same flat file space. If the file exists, it will be overwritten.

Saves the current radio configuration to a named configuration file. TXMission implements support for multiple radio configurations through an embedded database that can hold up to 100 named configurations. Configurations can be considered as virtual files and can be manipulated in a similar way to files within OOPS! but within the radio operating system are stored as records in a non-volatile database.

Example:

save myconfig

set service tx dvbs2

…

select myconfig

This saves the current radio settings to a named configuration and then modifies the current configuration to set the transmit path service to DVB-S2. At a later point the named configuration is loaded as the current configuration, meaning that any changes made in the meantime, such as to the transmit path service, are lost. See related configuration commands: *del*, *select*, *touch* and *dir*.

set carrier

set carrier (off | on)

Switches the transmit carrier off or on. The default setting is off.

Example:

set carrier on

set comp

set comp (off | on)

Switches H.264/H.265 image/video compression off or on. The default setting is off. Detection of the coding standard, frame rate, etc. is automatic.

Example:

set comp on

set freq

set freq [tx] [rx] <value>

Arguments:

tx	Applies the command to the transmit path of the radio.
rx	Applies the command to the receive path of the radio.
	If neither *tx* nor *rx* is specified then the command is applied to both paths.
<value>	Specifies the transmit or receive carrier center frequency in MHz. The range of supported frequencies is dependent on the RF capabilities of the radio. The default is 950.000000MHz.

Example:

set freq tx rx 2300.000000

set interface

set interface [tx] [rx] <value>

Arguments:

tx	Applies the command to the transmit path of the radio.
rx	Applies the command to the receive path of the radio.
	If neither *tx* nor *rx* is specified then the command is applied to both paths.
<value>	Specifies the physical interface to be used by the radio when transmitting or receiving data. Supported interfaces are: eth, usb, lvds, spacewire, rs485, sdi, uart, can, i2c, spi. The default is Ethernet.

Example:

set interface tx rx eth

sets the data interface to Ethernet for both the transmit and receive paths (transmit and receive could potentially use different interfaces).

set modcod

set modcod [tx] [rx] [--frame=(short | normal)] [--pilots=(off | on)] <value>

Options:

-f, --frame	Specifies the DVB-S2/S2X frame size. This option is not used with other services. The default is normal frames (these are higher latency but require less power).

-p, --pilots Switches DVB-S2/S2X pilots off or on. This option is not used with other services. Pilots allow a carrier to be received lower into the noise but reduce spectrally efficiency). The default is off.

Arguments:

tx Applies the command to the transmit path of the radio.

rx Applies the command to the receive path of the radio.

If neither *tx* nor *rx* is specified then the command is applied to both paths.

<value> Specifies the transmit or receive modulation and FEC rate. The default is QPSK ½ in all modes.

For DVB-S2X normal frames, the valid values are qpsk-13/45, qpsk-9/20, qpsk-11/20, 8psk-23/36, 8psk-25/36, 8psk-13/18, 8apskl-5/9, 8apskl-26/45, 16apsk-26/45, 16apsk-3/5, 16apsk-28/45, 16apsk-23/36, 16apsk-25/36, 16apsk-13/18, 16apsk-7/9, 16apsk-77/90, 16apskl-5/9, 16apskl-8/15, 16apskl-1/2, 16apskl-3/5, 16apskl-2/3, 32apsk-32/45, 32apsk-11/15, 32apsk-7/9, 32apskl-2/3, 64apsk-11/15, 64apsk-7/9, 64apsk-4/5, 64apsk-5/6, 64apskl-32/45.

For DVB-S2X short frames, the valid values are qpsk-11/45, qpsk-4/15, qpsk-14/45, qpsk-7/15, qpsk-8/15, qpsk-32/45, 8psk-7/15, 8psk-8/15, 8psk-26/45, 8psk-32/45, 16apsk-7/15, 16apsk-8/15, 16apsk-26/45, 16apsk-3/5, 16apsk-32/45, 32apsk-2/3, 32apsk-32/45.

For DVB-S2, the valid values are qpsk-1/4, qpsk-1/3, qpsk-2/5, qpsk-1/2, qpsk-3/5, qpsk-2/3, qpsk-3/4, qpsk-4/5, qpsk-5/6, qpsk-8/9, qpsk-9/10, 8psk-3/5, 8psk-2/3, 8psk-3/4, 8psk-5/6, 8psk-8/9, 8psk-9/10, 16apsk-2/3, 16apsk-3/4, 16apsk-4/5, 16apsk-5/6, 16apsk-8/9, 16apsk-9/10, 32apsk-3/4, 32apsk-4/5, 32apsk-5/6, 32apsk-8/9, 32apsk-9/10. (Note that rate 9/10 is not available when using DVB-S2 short frames.)

For CCSDS Viterbi/Reed-Solomon, the valid values are bpsk-1/2, bpsk-2/3, bpsk-3/4, bpsk-5/6, bpsk-7/8, qpsk-1/2, qpsk-2/3, qpsk-3/4, qpsk-5/6, qpsk-7/8, oqpsk-1/2, oqpsk-2/3, oqpsk-3/4, oqpsk-5/6, oqpsk-7/8.

Examples:

set modcod tx qpsk-1/2

set modcod rx -f=normal -p=off 8psk-3/4

set id

set id <value>

Arguments:

<value> A string that specifies a unique name for the equipment that is meaningful to the user (such as its location or service description). The factory default is an empty string.

Example:

set id "CubeSat 1 downlink"

set ip

set ip [[(eth0 | eth1) <address>] [subnet <address>]] [gw <address>]

Arguments:

eth0	Indicates the associated address is for the radio's monitor and control interface. The factory default is 192.168.123.1.
eth1	Indicates the associated address is for the radio's traffic interface. This does not have a factory default value.
<address>	A valid IPv4 address (e.g. 192.168.123.55).
subnet	Indicates that the associated address is a subnet mask (e.g. 255.255.255.0). Note that the use of CIDR (Classless Inter-Domain Routing) notation (e.g. /24) is not supported when defining addresses and subnets.
gw	Indicates that the associated address is a default gateway (e.g. 192.168.123.2).

This command is used to set the IP addresses for the radio's monitor and control Ethernet interface and traffic Ethernet interface. The radio operates as a Layer 2 switch and does not require the traffic interface to be assigned an address. A monitor and control address is essential for accessing the built-in web server (and for issuing OOPS! commands through a telnet session). It is common practice for the traffic function and the monitor and control function to be on different subnets in order to create traffic separation. Although the command allows a gateway to be set for both addresses, there is only one TCP/IP stack in the radio and therefore only one gateway, therefore the gateway is the same in both cases. If a second gateway is required, this can be added as a route via the operating system (please contact us for further details). Although the command only accepts IPv4 addresses, all IPv6 packets on the network will be correctly bridged through the radio.

Examples:

set ip eth0 10.0.0.1

set ip eth0 10.0.0.1 subnet 255.255.255.0 gw 10.0.0.2

set ip gw 10.0.0.2

set power

set power <value>

Arguments:

<value>	Specifies the modem transmit power level prior to any RF amplification. The range of supported values is -40.0 to +5.0dBm. The default is -40.0dBm.

Example:

set power -22.3

set rate

set rate [txdata | txsym] [rxdata | rxsym] <value>

Arguments:

txdata	Applies the command to the transmit data rate.
txsym	Applies the command to the transmit symbol rate.
rxdata	Applies the command to the receive data rate.
rxsym	Applies the command to the receive symbol rate.
<value>	Specifies the transmit or receive data rate or symbol rate.

The range of supported data and symbol rates is hardware, service and modcod specific. The default in each case is the lowest supported data rate or symbol rate. Note that transmit and receive data rates or symbol rates can be set simultaneously (providing they require the same value) but data and symbol rates cannot be specified together in one command. At least one optional parameter must be specified. Note that there is a fixed relationship between data rate and symbol rate and changing either will change the other regardless of what prior commands have been issued. (Some users will prefer to work in symbols and others in bits.)

Examples:

set rate txdata rxdata 10.000000

sets both the transmit and receive paths to 10Mbps.

set rate rxsym 5.000000

sets the receive symbol rate to 5Mbps.

set rolloff

set rolloff [tx] [rx] <value>

Arguments:

tx	Applies the command to the transmit path of the radio.
rx	Applies the command to the receive path of the radio.
	If neither *tx* nor *rx* is specified then the command is applied to both paths.
<value>	Specifies the transmit or receive carrier roll-off factor, which should be one of the following values: 5, 10, 15, 20, 25, 35, 40.

Example:

set rolloff 35

sets both the transmit and receive roll-off factors to 35%.

set service

set service [tx] [rx] <mode> [<submode>]

Arguments:

tx — Applies the command to the transmit path of the radio.

rx — Applies the command to the receive path of the radio.

If neither *tx* nor *rx* is specified then the command is applied to both paths.

<mode> — Specifies the transmit or receive service. This is one of: off, *dvbs2*, *dvbs2x*, *ccsds-dvbs2*, *ccsds-dvbs2x*, *ccsds-vit*, *ccsds-vit-rs*. Transmit and receive services are completely independent of each other. The factory default is for both services to be switched off.

<submode> — For DVB-S2/S2X services, this specifies whether the modcod(s) being generated with respect to the carrier should be static or change dynamically. The options are *ccm*, *vcm* and *acm*. These terms are defined in the relevant DVB-S2/S2X standards. The factory default is CCM mode.

In CCM mode, the carrier is generated using a fixed modcod that never varies unless changed by the user. VCM mode is similar to the description of ACM below but with the distinction that while modcods can be continuously changed without causing the receiver to unlock, there is no feedback path from the receiver to the transmitter that determines modcod selection. Instead, any change of modcod is determined programmatically at the transmitter (i.e. by the user application).

In ACM mode, the modcod being generated at any point in time is varied continuously to maximise the data rate for the strength of signal at the receiver. For a LEO/MEO satellite, the strength of signal received at any ground station will vary during each pass. If a fixed modcod is used then this is typically selected to allow the ground station to lock to the received signal shortly after the satellite appears on the horizon. By the time the satellite is overhead the received power level will be much higher. ACM makes use of the variable signal strength by dynamically selecting a modcod that provides just enough error protection to preserve the quality of service. Reducing or increasing the number of redundancy bits in the transmitted output (used for error correction) causes a corresponding inverse increase or decrease in the number of useful data bits that are transmitted. A higher signal strength also allows a higher order modulation to be used (which has more bits per symbol and is therefore more spectrally efficient).

ACM expects a feedback path to exist between the receiver and transmitter in order to pass information pertaining to the current signal strength conditions being experienced at the receiver. However, it is also possible to put the receiver into either VCM or ACM mode (so that it accepts whatever modcod it receives) and for the satellite to change the transmitted modcod according to the satellite's position relative to the ground. ACM mode has been designed to ensure no loss of data in either the transmitter or receiver regardless of the frequency of changes to the modcod.

The amount of data that can be downloaded per satellite pass when using ACM mode is typically several times that of CCM mode.

Example:

set service tx dvbs2

select

select <filename>

Arguments:

<filename> Specifies the name for an existing radio configuration.

This reconfigures the radio using the contents of the named configuration. Note that any commands that are subsequently issued are applied to the current configuration, not the file from which the current configuration was taken (unless the redirection operator is used to redirect the commands to the relevant file). See related configuration commands: *touch*, *dir*, *save* and *del*.

Example:

select myconfig

submit

submit

When using *batch* mode, all commands that are received by the radio are held pending until a command is received that has a *submit* suffix (for example, *set carrier on submit*) or an explicit *submit* command is issued. The use of *batch* mode and the *submit* command avoids reconfiguration occurring in a piecemeal fashion. If *batch* mode is not enabled when a *submit* command is issued, the command has no effect. Likewise, if no commands are pending at the time a *submit* command is issued.

touch

touch <filename>

Arguments:

<filename> Specifies the name for a new radio configuration. Valid names are as per Linux filenames. There is no concept of nested folders and all configurations exist in the same flat file space. On file creation, the configuration defaults all radio settings to their factory defaults.

TXMission implements support for multiple radio configurations through an embedded database that can hold up to 100 named configurations. Configurations can be considered as virtual files and can be manipulated in a similar way to files within OOPS! but within the radio operating system are actually stored as records in a non-volatile database. Once a configuration has been created then the contents can be changed using the redirection operator '>' with any regular command. The database itself can be

manipulated through standard Linux and FTP commands (allowing, for example, all configurations to be copied from one radio to another). Please contact TXMission for further details.

Example:

touch myconfig

set service tx dvbs2 > myconfig

select myconfig

This creates a new configuration and then sets the transmit path service in that configuration to DVB-S2 before finally applying the configuration to the radio, replacing its current configuration. See related configuration commands: *del*, *select*, *save* and *dir*.

Appendix A: TXMission Platform Performance Characteristics

Platform	Parameter	Performance
Quest Hardware Platform A/B	Data rate in DVB-S2/S2X modes	0.050000 to 500.000000Mbps
Quest Hardware Platform A/B	Data rate in Viterbi-RS modes	0.009600 to 50.000000Mbps
Connect Hardware Platform	Data rate in DVB-S2/S2X modes	0.050000 to 500.000000Mbps
Connect Hardware Platform	Data rate in Viterbi-RS modes	0.009600 to 50.000000Mbps
Quest Hardware Platform A/B	Symbol rate in DVB-S2/S2X modes	0.100000 to 125.000000Msps
Quest Hardware Platform A/B	Symbol rate in Viterbi-RS modes	0.009600 to 40.000000Msps
Connect Hardware Platform	Symbol rate in DVB-S2/S2X modes	0.100000 to 125.000000Msps
Connect Hardware Platform	Symbol rate in Viterbi-RS modes	0.009600 to 40.000000Msps
Quest/Connect S-band RF	Frequency range	0.000000 to 6000.000000MHz
Quest/Connect X-band RF	Frequency range (uplink)	7200.000000 to 7750.000000MHz
Quest/Connect X-band RF	Frequency range (downlink)	8000.000000 to 8400.000000MHz

