DRAFT - To Be Confirmed

Elysium Radio User Manual

Andrew Wygle

Adamant Aerospace

July 24, 2017

Abstract

The Elysium VHF/UHF Radio is designed for Telemetry, Tracking, and Command (TT&C) applications for high-performance CubeSats. Its low power consumption, full-duplex architecture, and high output power simplify on-orbit operations while maximizing the probability of mission success.

Receiver *active* power consumption is less than 25 mW, allowing the radio to remain powered at all times without impacting the overall mission power budget. Full-duplex capability ensures that all command packets arrive safely regardless of telemetry downlink.

The radio can be tuned from 137 MHz to 1 GHz, allowing the use of all available VHF or UHF satellite bands and providing robustness against regulatory uncertainty.

A high-efficiency, scalable power amplifier allows maximum RF output power to be tailored from 1 to 5 Watts at 50% or better efficiency, enabling the use of low-cost antennas or high data rates as required and reducing the amplifier's thermal dissipation requirements compared to traditional linear amplifiers.

These traits and others make the Elysium radio ideally suited to serve as a primary spacecraft radio or as a crucial backup to a high data rate S- or X-band system.

Contents

List of Figures

List of Tables

1 Overview

The Elysium VHF/UHF Radio is a low-power, high-efficiency FSK modem for satellite communications. The Elysium delivers high performance at much lower currents than traditional CubeSat radio solutions. The use of standard data link protocols and modulation schemes simplifies integration with existing ground station networks. The small form factor allows the maximum possible amount of space to be reserved for the primary payload, while the available CubeSat Kit Adapater provides compatibility with existing CubeSat hardware from vendors like Pumpkin, GomSpace, and Clyde Space.

This section provides an overview of the Elysium radio. Section 2 provides detailed information on the mechanical and thermal design of the radio. Section 3 dives deeply into the electrical interface of the system. Section 5 gives an overview of the command, configuration, and telemetry interfaces, which are dicussed in more detail in Section 7, Section 8, Section 9, Section 10, and Section 11.

2 Mechanical

2.1 Overview

The Elysium PCB is a small, 4-layer board made primarily of Isola FR408 high-speed laminate. It is designed to take up as little room as possible, so that it can be located as close as possible to the spacecraft antenna for maximum performance. An adapter board is also available which makes the Elysium compatible with the CubeSat Kit Bus, an electrical standard used by many existing CubeSat component manufacturers.

2.2 Mounting

The Elysium is designed with 4 board-level mounting holes spaced 40mm apart around the center of the board. These mounting holes are 2.4mm clearance holes, suitable for M2 fasteners. #1 and #2 fasteners may also be used depending on the design of the spacecraft structure. The mounting hole copper pads are connected directly to the Elysium electrical ground plane.

In addition to the board level mounting holes, there are two mounting holes provided for attaching a heat sink, which are 1.8mm clearance holes suitable for M1.6 fasteners. #0 fasteners may also be used depending on heat sink design.

See Figure 1 for a drawing of the mounting interface.

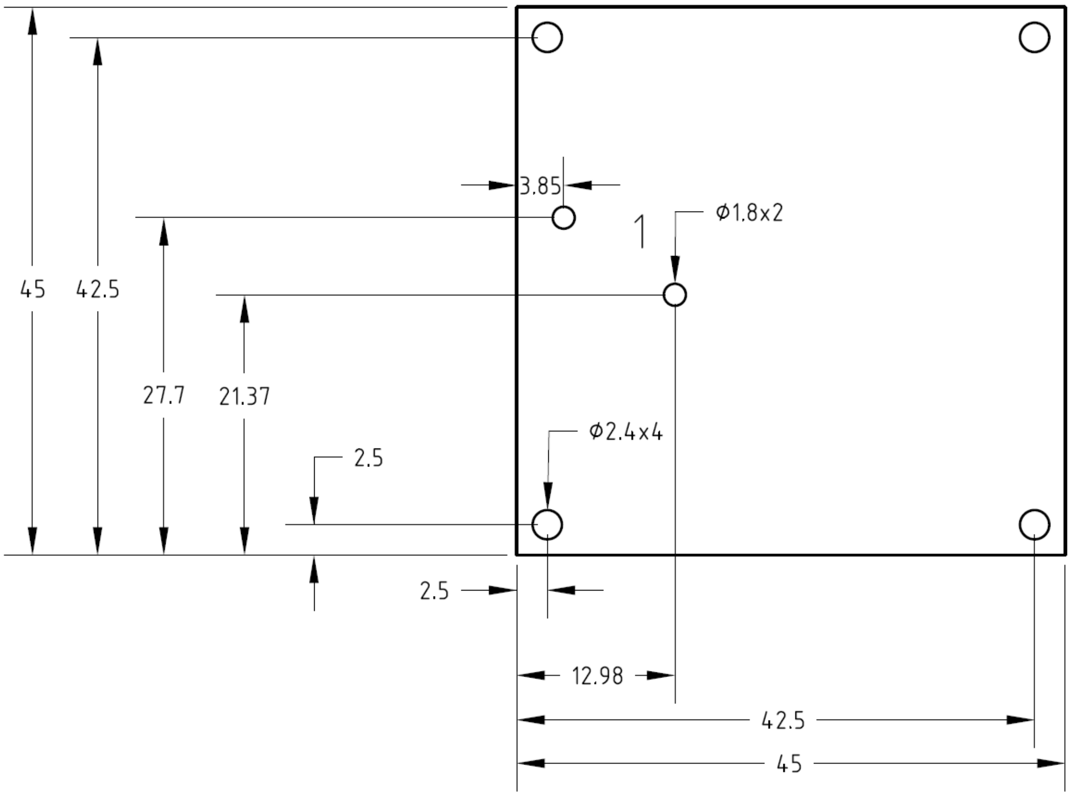


Figure 1: Elysium Mounting Holes
All dimensions in mm

2.3 Bounding Envelope

The Elysium PCB is a 45mm square, however the SMA connectors which connect to the spacecraft's antennae extend beyond this envelope. See Figure 2 and Figure 3 for details of the SMA bounding envelope.

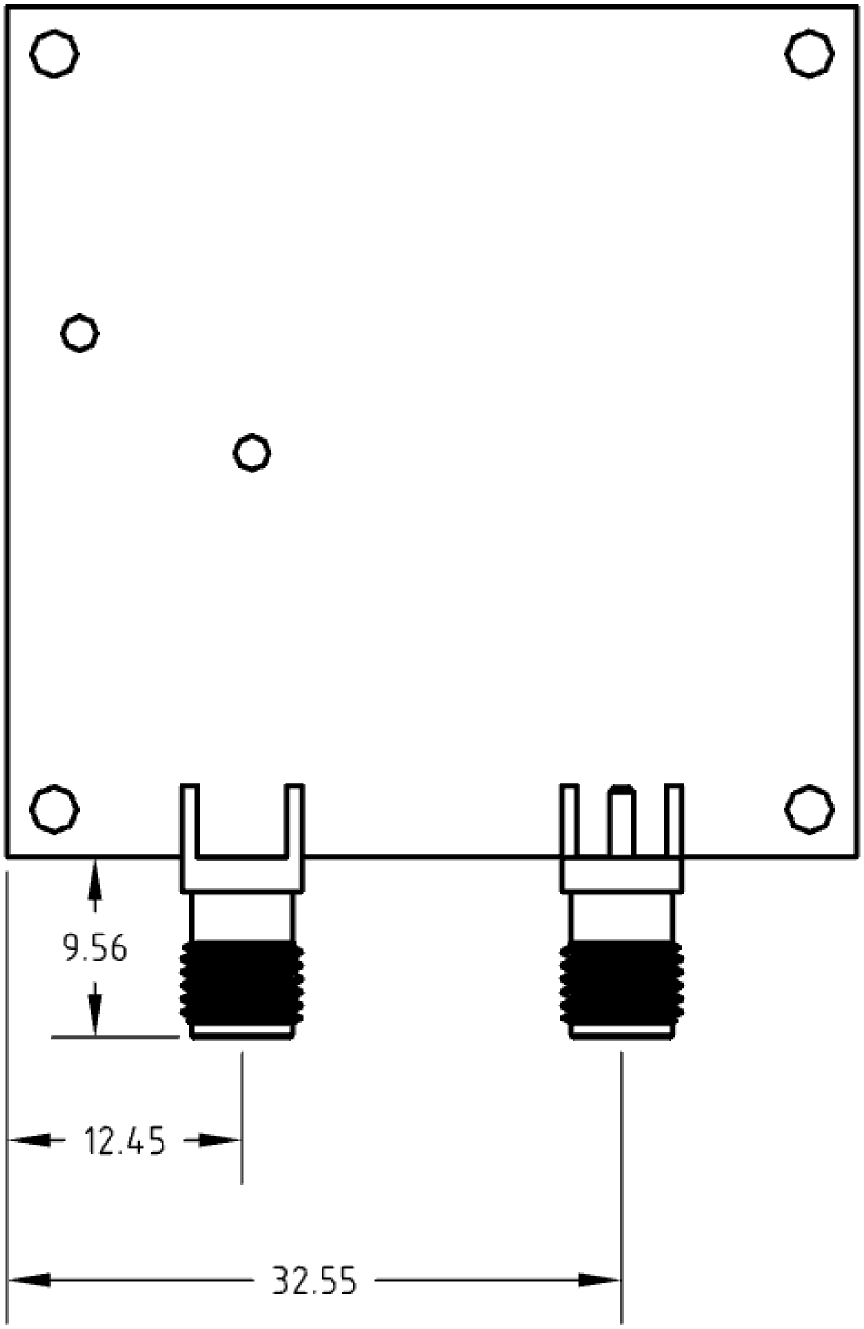


Figure 2: SMA Connectors Top View
All dimensions in mm

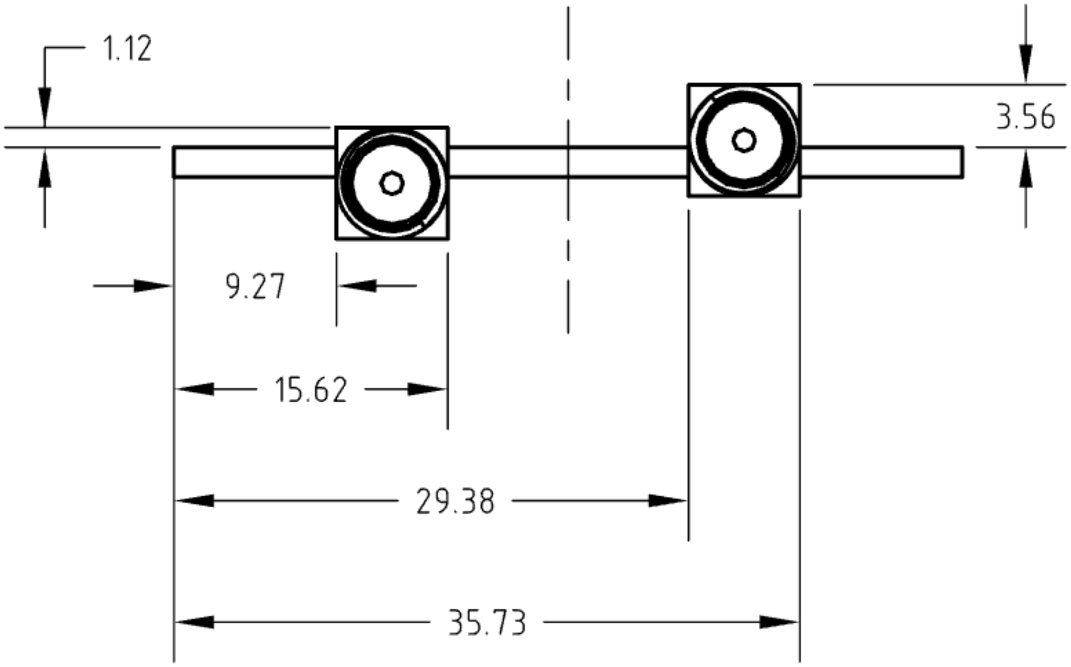


Figure 3: SMA Connectors Front View
All dimensions in mm

The thickness of the board is 1.575mm or 62mil. Component heights do not exceed 2mm, however both sides of the radio require board-level shielding with a height of 3mm on a side for a total height of 7.575mm for the module. See Figure 4, Figure 5, and Figure 6 for detailed drawings of these elements.

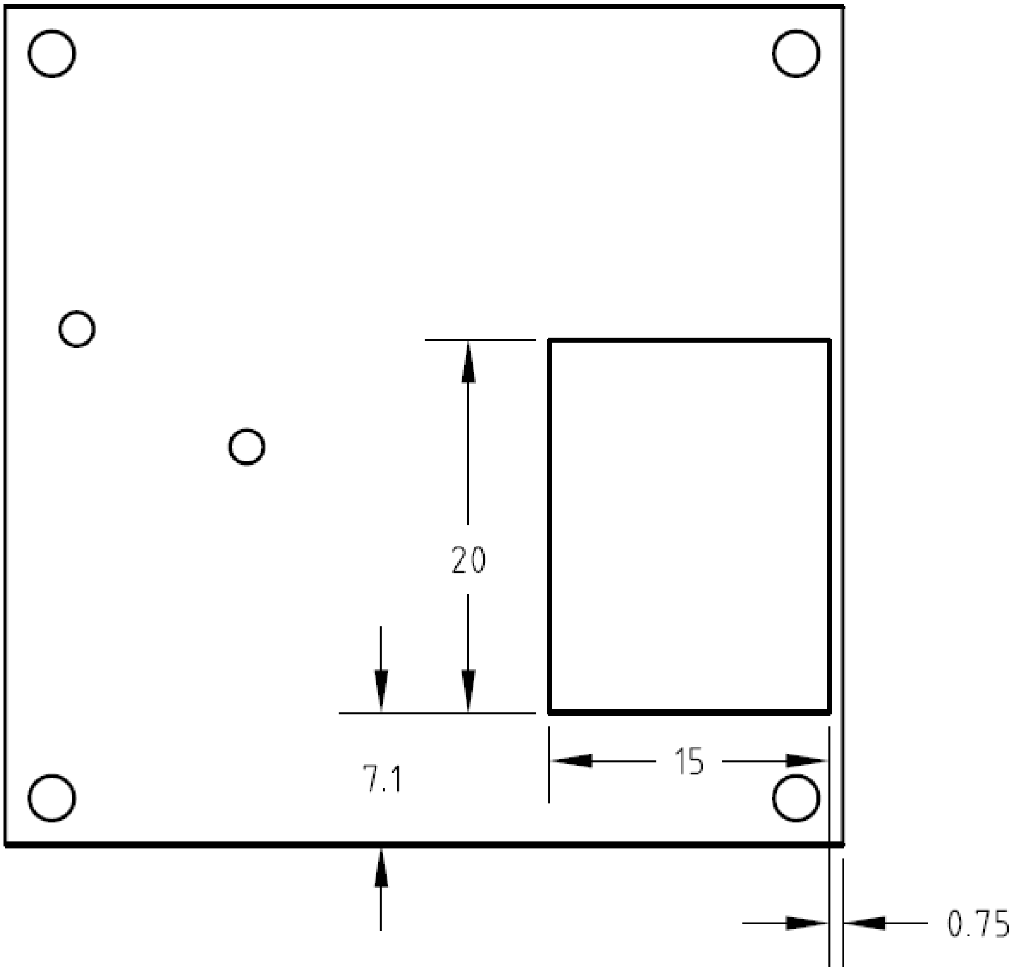


Figure 4: RX Shield Location
All dimensions in mm

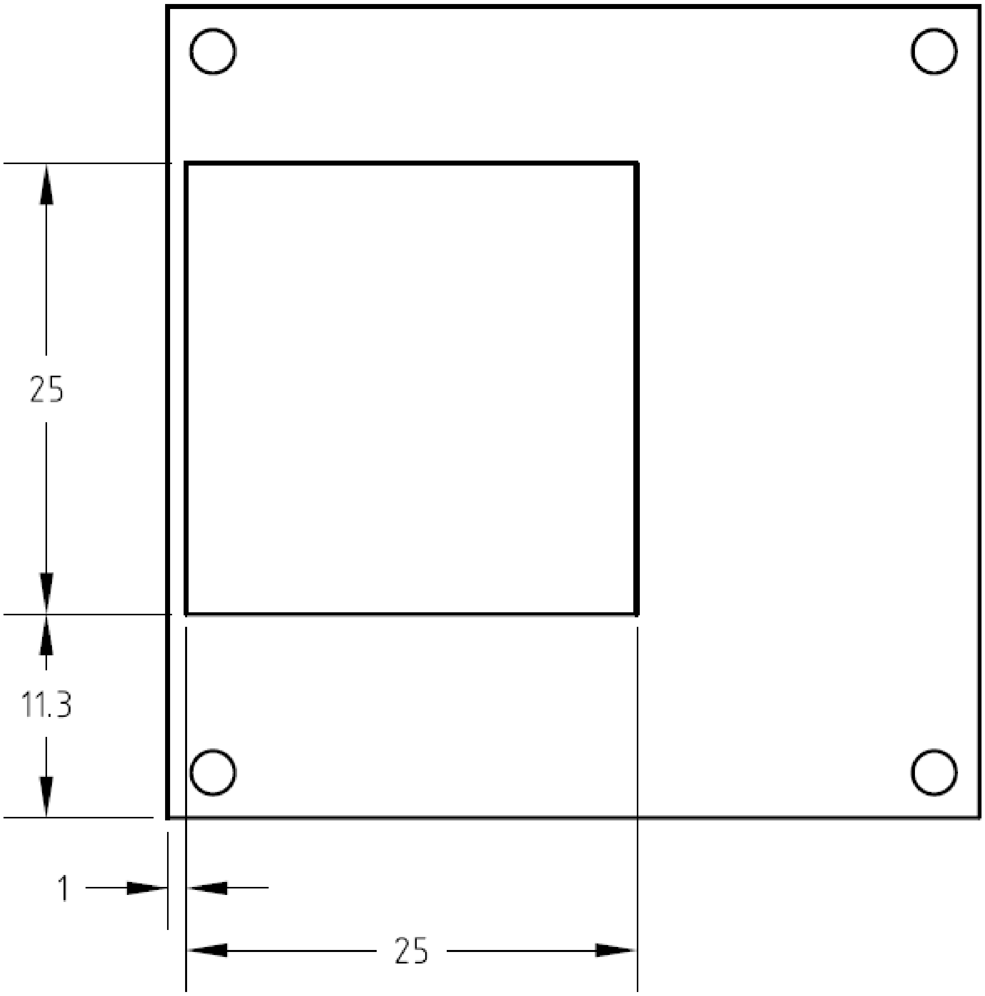


Figure 5: TX Shield Location
All dimensions in mm

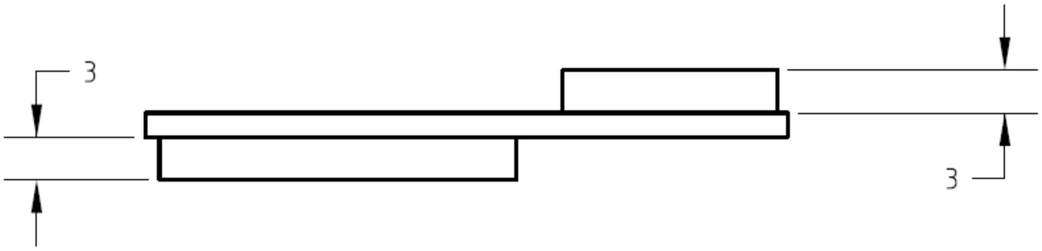


Figure 6: Shielding Thicknesses
All dimensions in mm

2.4 Thermal

The Elysium is designed for an operating ambient temperature range of -40 to +85 $^{\circ}\mathrm{C}$. However, the RF power amplifier has additional requirements for thermal dissipation. The junction of the amplifier must be maintained at no more than $150\,^{\circ}\mathrm{C}$, while the package resistance to the thermal pad is $11\,^{\circ}\mathrm{C}/\mathrm{W}$. Based on the $>$50% efficiency of the amplifier at saturation, an analysis of heat sink requirements can be performed in the context of the mission's overall thermal environment.

The thermal interface to the heat sink may be provided via an exposed copper pad on the top side of the board with dimensions 15x13mm. This copper pad is coupled to the ground pad of the power amplifier through 35 0.254mm thermal vias with 2.5-oz copper plating, plugged with non-conductive and plated shut. See Figure 7 for a cross-section view of the thermal vias.

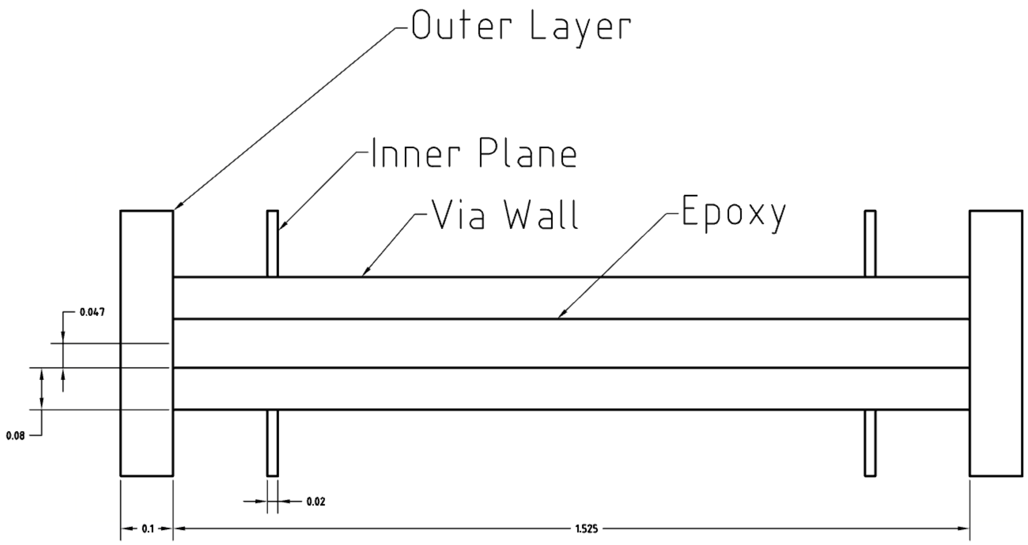


Figure 7: Thermal Via Cross-Section
All dimensions in mm

The thermal resistance from the ground pad of the power amplifier to the exposed copper heat sink interface pad is estimated to be $2\,^{\circ}\mathrm{C}/\mathrm{W}$. This value can be measured through the use of a pair of thermistors (Panasonic ERT-J0EP473F) which are optionally present on either side of the PCB as near as possible to the power amplifier and which can be read through the radio's Channel interface to on board telemetry. The heat sink will require a cutout if the thermistors are to be used in flight.

It is also possible to design a custom heat sink which makes contact with the top of the power amplifier package. If this is intended, please contact Adamant prior to your purchase to discuss the best way to mitigate EMI/RFI concerns.

See Figure 8 for more details of the heat sink interface.

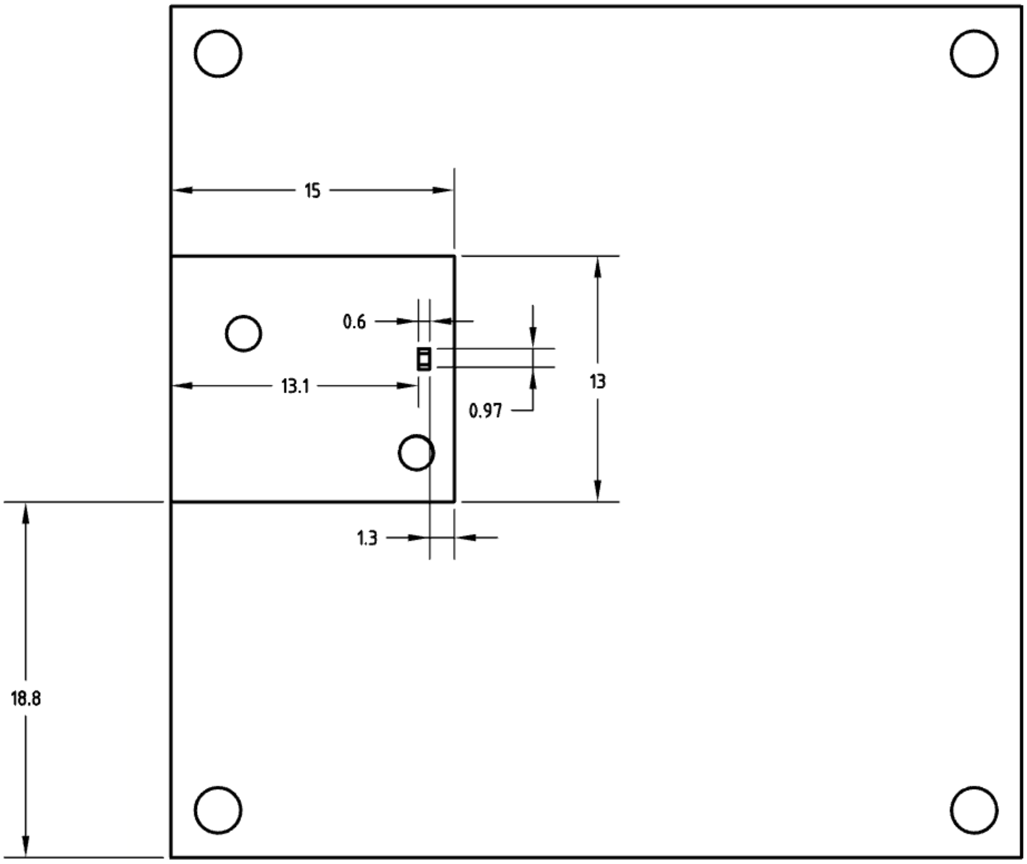


Figure 8: Heatsink Interface Detail View
All dimensions in mm

3 Electrical

3.1 Pinout

The Elysium uses a standard 100-mil header connector (Samtec TSW-106-07-G) for its electrical interface. The pinout is described in Table 1.

An additional interface compatible with board-to-board one-piece compression connectors such as the Samtec SIB-106-02-F-S is also present on the bottom of the board and has the same pinout. This interface is used in the CubeSat Kit Adapter board to allow compliance with the PC-104 mechanical envelope.

A three-pin interface compatible with press-fit 100-mil header connectors such as the Samtec PHT-103-

Table 1: Elysium Pinout

Pin #	*Net Name*	*Description*	*Notes*
1	GND	Electrical Ground Connection	
2	GPO	General-purpose output to downstream devices	
3	5V	Electrical Power Connection	$5V \pm 5\%$
4	TX	UART Data Connection TX Line	Input (DCE)
5	RX	UART Data Connection RX Line	Output (DCE)
6	$\overline{\text{RESET}}$	Elysium Reset Line Input	Active Low

Table 2: Programming Header Pinout

Pin #	*Net Name*	*Description*	*Notes*
1	SBWTDIO	SBW Data Line	Bi-Directional
2	SBWTCK	SBW Clock Line	Max Frequency 10 MHz
3	GND	Electrical Ground Connection	

01-S-S is used for programming the on-board microcontroller using the Spy-Bi-Wire interface. Its pinout is described in Table 2.

> NOTE: The programming header is needed only for initial firmware load, custom firmware development, and disaster recovery. In other circumstances, the in-system or over-the-air programming capabilities described in Section 7 should be preferred.

3.2 Electrical Characteristics

The voltage, current, and frequency characteristics of all interface signals are defined in Table 3.

4 RF

4.1 Interface

The Elysium uses two 50-Ohm SMA connectors, Samtec SMA-J-P-H-ST-EM1, as its RF interface connectors. Connector J1, located on the top of the board, is the receiver interface, while Connector J2 on the bottom of the board is transmitter interface.

The transmit connector J2 is AC-coupled to the RF circuitry using DC-blocking capacitors. The receive connector J1 supports a DC bias of $3.0\,\mathrm{V}$ through the use of a bias tee. This DC bias is intended to power an off board LNA, as described in Section 4.3.2. If the DC bias is not applied, then connector J1 is also AC-coupled.

Table 3: Electrical Characteristics

Absolute Maximum Ratings				
	Min	*Max*	*Unit*	*Notes*
V_{cc}	-0.3	6	V	
I_{cc}		3	A	
V_i Input Voltage	-0.3	7	V	Primary Header
	-0.3	2.8	V	Programming Header
Recommended Operating Conditions				
	Min	*Max*	*Unit*	*Notes*
V_{cc}	4.75	5.25	V	
V_i Input Voltage	0	5.25	V	Primary Header
	0	2.8	V	Programming Header
Operating Temperature	-40	+85	C	See Section 2.4
Electrical Characteristics				
	Min	*Max*	*Unit*	*Notes*
V_{IH}	2.02		V	Primary Header
	1.375	1.875	V	Programming Header
V_{IL}		0.8	V	Primary Header
	0.625	1.125	V	Programming Header
Timing Characteristics				
	Min	*Max*	*Unit*	*Notes*
f_{UART}		4	MBaud	TX & RX
T_{RST} $\overline{\text{RESET}}$ pulse duration	2		µs	

Table 4: Modulator Characteristics

Value	*Description*	*Conditions*	*Min*	*Typ*	*Max*	*Unit*
$\mathrm{BR}_{\mathrm{TX}}$	Transmitter Bit Rate	FSK Modulations	1.2		300	kbps
$\mathrm{F}_{\mathrm{DEV}}$	Frequency Deviation from Carrier		0.6		200	kHz
BT	Bandwidth-Time Product	Fixed Options	0.3	0.5	1.0	

4.2 Transmitter

The Elysium uses a high-efficiency Class C amplifier combined with a state-of-the-art transmitter to provide high output power at high data rates for minimum power consumption.

4.2.1 Modulation

The Elysium supports transmitting binary FSK, with programmable frequency deviation/modulation index. An optional Gaussian filter may also be used with three distinct bandwidth-time products. The combination of these factors allows the Elysium to support 2-FSK, 2-MSK, 2-GFSK, and 2-GMSK modulations.

Certain restrictions apply - the entire modulated signal must fit within the 250 kHz bandwidth of the modulator. This means that the condition $F_{DEV} + (BR_{TX}/2) \leq 250\,\mathrm{kHz}$, where $\mathrm{F}_{\mathrm{DEV}}$ is the deviation of the two signal tones from the carrier frequency (i.e. $\frac{|f_{mark} - f space|}{2}$) and $\mathrm{BR}_{\mathrm{TX}}$ is the bit rate, must be satisfied for correct operation of the modulator.

The characteristics of the Elysium's modulator are listed in Table 4.

The radio also supports an experimental spread-spectrum modulation for use in environments with a very high probability of interference from transmitters operating in the same band. Please contact Adamant for more information.

4.2.2 Frequency Bands

The transmit chain of the radio can be tuned over a wide frequency band, from 137 MHz on the low end to 1 GHz on the high end. However, the entire band is not available for a single mission. Physical changes to the installed components on the board create a narrower frequency band of operation. Several such bands have been tested by Adamant to ensure performance, however if your mission requires an operating frequency outside of these bands please contact us and we can work to define new frequency bands as needed. The presently supported frequency bands are defined in Table 5.

4.2.3 Output Power

In order to achieve efficiencies over 50%, the amplifier must be operated in the saturated mode. By changing the components installed on the PCB, we can change the bias point of the amplifier and thus the output power at saturation. This defines the maximum output power of the amplifier. However, output power may

Table 5: Supported Transmit Frequency Bands

Min	*Max*	*Unit*
137	175	MHz
433	470	MHz
865	955	MHz

Table 6: Demodulator Characteristics

Value	*Description*	*Conditions*	*Min*	*Typ*	*Max*	*Unit*
BR_{RX}	Receiver Bit Rate	310 to $680\,\text{MHz}$	0.78		150	kbps
		680 to $960\,\text{MHz}$	1.56		200	kbps
F_{DEV}	Frequency Deviation from Carrier		33		200	kHz
BW_{RX}	Receive Filter Single-Sided Bandwidth	25 kHz Steps	25		400	

be commanded to as much as 19 dB less than this value. E.g., when the amplifier is configured for $4\,\text{W}$ maximum output $(36\,\text{dBm})$, the user may command the radio to transmit at $2\,\text{W}$ of output power $(33\,\text{dBm})$ instead, but should not expect an efficiency of 50% in such a case. The minimum output power in that case would be $50\,\text{mW}$ $(17\,\text{dBm}$, or $36\text{-}19\,\text{dBm})$. Note that overall DC power consumption does decrease as commanded output power decreases. The amplifier's saturation point may be configured in $1\,\text{W}$ steps from $1\,\text{W}$ to $5\,\text{W}$.

4.3 Receiver

4.3.1 Demodulation

The Elysium supports receiving binary FSK modulation with programmable frequency deviation/modulation index. Due to choices made to reduce receiver power consumption, the receiver is most sensitive to signals with modulation index $\beta \geq 2$.

The characteristics of the Elysium's demodulator are listed in Table 6. Information on noise figure and sensitivity over frequency bands can be found in Section 4.3.3.

4.3.2 Low-Noise Amplifier

Adamant also produces a discrete low-noise amplifier for use with the Elysium radio. It typically improves receiver sensitivity by 3-5 dB in exchange for dissipating 10-15 mW of additional power.

The low-noise amplifier is powered by an integrated bias tee on-board the Elysium. If the bias supply is enabled, the RX port will have a DC bias of 3.0 V nominal applied to power the low-noise amplifier. While this is perfectly safe under most operating conditions, certain antenna configurations may require a zero

DC bias. Under those circumstances, if not using the LNA, the bias tee may be deactivated through a configuration option at the time of a firmware build. The LNA input has zero DC bias.

Receiver performance with and without the use of the LNA is listed in Table 7. Contact Adamant for more details.

4.3.3 Frequency Bands

The receive chain of the radio can be tuned over a wide frequency band, from 300 MHz on the low end to 960 MHz on the high end. However, the entire band is not available for a single mission. Physical changes to the installed components on the board create a narrower frequency band of operation. Several such bands have been tested by Adamant to ensure performance, however if your mission requires an operating frequency outside of these bands please contact us and we can work to define new frequency bands as needed. The presently supported frequency bands are defined in Table 7.

5 Software

5.1 Overview

The Elysium radio firmware makes use of five basic concepts - Registers, Commands, Channels, Errors, and Events. Each of these concepts serves a specific role when interacting with the radio.

Registers store configuration information for the radio. Each register has an 8-bit address and stores an 8-bit data value, although some registers are grouped together into larger conceptual units (e.g. 32-bit values) and others are broken up into smaller units (e.g. bitfields). More details on Registers can be found in Section 8.

Commands instruct the radio to take some action, such as modifying a register value or subscribing to an Event notification. Each Command has an ID which is part of a common 8-bit ID space shared by Commands, Channels, Errors, and Events. Command IDs occupy the range from 0x00 to 0x3F. More details on Commands can be found in Section 7.

Channels provide a synchronous reporting and logging mechanism for data, such as temperature measurements, which changes regularly over time. Channel data may be reported out to other elements of the spacecraft network, or may be logged to on-board telemetry storage for later downlink. Each Channel has an ID which is part of a common 8-bit ID space shared by Commands, Channels, Errors, and Events. Channel IDs occupy the range from 0x40 to 0x7F. Each Channel also has an associated Data Type, which specifies the kind of values the Channel produces. More information on Channels can be found in Section 9.

Errors provide an asynchronous reporting mechanism for anomalous events, such as an overtemperature or undervoltage. Error reports have an associated priority, from DEBUG to CRITICAL, which can often be configured through a particular Register. Errors may be reported to a designated member of the spacecraft network or logged as telemetry for later downlink. Reporting and logging of Errors may be filtered based on priority. Each Error has an ID which is part of a common 8-bit ID space shared by Commands, Channels,

Table 7: Supported Receive Frequency Bands

431.5 *to* $436.5\,\mathrm{MHz}$					
	Min	*Typ*	*Max*	*Unit*	*Notes*
Sensitivity		-104		dBm	25 kbps, 100 kHz
		-101		dBm	66.7 kbps, 200 kHz
		-98		dBm	150 kbps, 300 kHz
w/ LNA		-109		dBm	25 kbps, 100 kHz
		-106		dBm	66.7 kbps, 200 kHz
		-104		dBm	150 kbps, 300 kHz
NF		7.0		dB	
w/ LNA		1.8		dB	Configuration-dependent
860.5 *to* $877.5\,\mathrm{MHz}$					
	Min	*Typ*	*Max*	*Unit*	*Notes*
Sensitivity		-104		dBm	25 kbps, 100 kHz
		-103		dBm	66.7 kbps, 200 kHz
		-99		dBm	150 kbps, 300 kHz
w/ LNA		-109		dBm	25 kbps, 100 kHz
		-106		dBm	66.7 kbps, 200 kHz
		-104		dBm	150 kbps, 300 kHz
NF		7.0		dB	
w/ LNA		2.0		dB	Configuration-dependent
902 *to* $928\,\mathrm{MHz}$					
	Min	*Typ*	*Max*	*Unit*	*Notes*
Sensitivity		-104		dBm	25 kbps, 100 kHz
		-101		dBm	66.7 kbps, 200 kHz
		-98		dBm	150 kbps, 300 kHz
w/ LNA		-109		dBm	25 kbps, 100 kHz
		-106		dBm	66.7 kbps, 200 kHz
		-104		dBm	150 kbps, 300 kHz
NF		7.0		dB	
w/ LNA		2.0		dB	Configuration-dependent

Errors, and Events. Error IDs occupy the range from 0x80 to 0xBF. More information on Errors can be found in Section 10.

Events provide an asynchronous reporting mechanism for any kind of event, such as a packet being received. Unlike Error reports, Event notifications have no associated priority. Event notifications may be subscribed to by any member of the spacecraft network or logged as telemetry for later downlink. Each Event has an ID which is part of a common 8-bit ID space shared by Commands, Channels, Errors, and Events. Event IDs occupy the range from 0xC0 to 0xFF. More information on Events can be found in Section 11.

Finally, the Elysium's primary responsibility is the passing of traffic from the spacecraft to the ground station and vice versa. The details of this networking responsibility can be found in Section 5.2, as well as in supporting documents.

5.2 Elysium Networking

The Elysium provides three layers of the OSI Basic Reference Model for networking - the Physical Layer, the Data Link Layer, and the Network Layer. The Physical Layer is provided by the radio hardware as described above. The Data Link and Network layers are provided in software.

The overall architecture of the Elysium networking system is displayed in Figure 9.

Data received over the UART interface is passed to the Network Layer subsystem for routing. The Network Layer subsystem examines the packet according to the standards of the routing protocol in use and determines whether the packet is intended for the Elysium firmware or to be routed over the radio link. Some basic integrity checks may also be performed at this stage.

Data not intended for the Elysium firmware is passed to the Data Link Layer subsystem where packets are broken apart if necessary and packed into frames for transmission over the radio link. These frames are then optionally passed to the Coding sublayer for Forward Error Correction before being sent over the radio link.

In the reverse direction, FEC decoding and packet reassembly are performed at the Data Link Layer subsystem before the Network Layer subsystem routes the received packets to the Elysium firmware or out the UART interface as necessary.

5.2.1 Network Layer

The Network Layer is responsible for doing the minimal amount of routing necessary for determining the destination of a packet of data.

Several versions of the Network Layer subsystem exist depending on the network layer protocol in use. Their details are explained in the Network Layer definition documents found on the Elysium page of the Adamant website. If your mission requires a network layer protocol which is not listed on the website, please contact Adamant to discuss adding support for it.

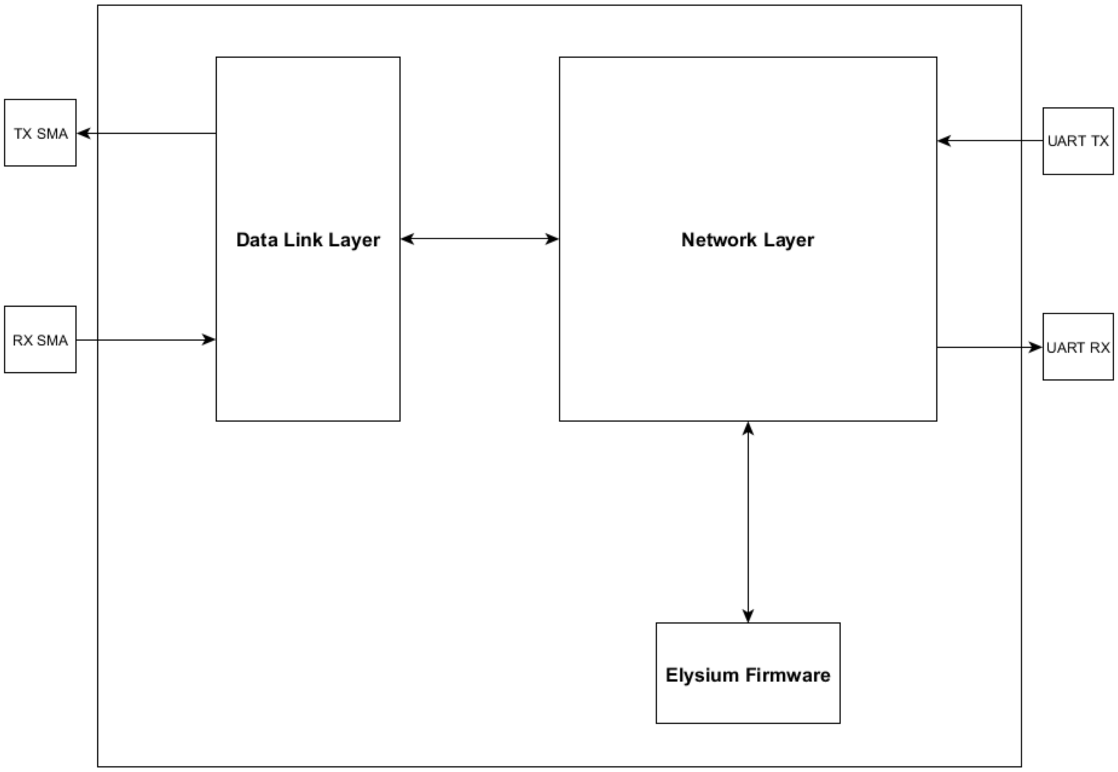


Figure 9: Networking Block Diagram

There are three potential destinations for a packet of data - the UART interface, the RF interface, or the Elysium firmware. Routing decisions among these three destinations are made according to the standards of the Network Layer subsystem in use.

The Network Layer is also responsible for translating the addresses of the network layer protocol in use into the 16-bit addresses used by the Elysium firmware, and vice versa. The exact rules for this translation can be found in the Network Layer definition documents.

The portion of the 8-bit register address space from address 0x80 to address 0xBF is reserved for registers related to Network Layer subsystems. The precise contents of these registers are defined in the Network Layer definition documents.

Portions of the 8-bit ID space are reserved for Channels, Errors, and Events related to Network Layer subsystems. These portions are:

- Channels: 0x60-0x6F
- Errors: 0xA0-0xAF
- Events: 0xE0-0xEF

Network Layer subsystems do not define additional commands.

5.2.2 Data Link Layer

The Data Link Layer subsystem is responsible for dividing the packets of the Network Layer into frames of the Data Link Layer in preparation for transport over the radio link, or reassembling divided packets after reception from the radio link. The Coding sublayer may also perform Forward Error Correction (FEC) encoding and decoding, if required. Certain implementations of the Data Link Layer subsystem may also provide Quality of Service (QoS) or Automatic Repeat Request (ARQ) features.

At present, only the CCSDS Telemetry (TM) and Telecommand (TC) Space Data Link Protocols (CCSDS Recommended Standards 132.0-B-2 and 232.0-B-3 respectively) are supported. The SDLP Data Link Layer's details are explained in the Data Link Layer definition document found on the Elysium page of the Adamant website. If your mission requires a different Data Link Layer protocol be used, please contact Adamant to discuss adding support for it.

The portion of the 8-bit register address space from address 0xC0 to address 0xFF is reserved for registers related to Data Link Layer subsystems. the precise contents of these registers are defined in the Data Link Layer definition documents.

Portions of the 8-bit ID space are reserved for Channels, Errors, and Events related to Data Link Layer subsystems. These portions are:

- Channels: 0x70-0x7F
- Errors: 0xB0-0xBF
- Events: 0xF0-0xFF

Data Link Layer subsystems do not define additional commands.

6 Applications

This section is dedicated to explaining how to accomplish certain common operational scenarios using the Elysium. Links to more detailed documentation will be present throughout the text.

6.1 Modifying Configuration

Matthew, a mission operator for a 3U CubeSat flying the Elyisum radio, needs to modify the configuration of the radio to add settings to match a new ground station. This new ground station does not support convolutional coding, but as a very large dish, should still provide reasonable link margin.

First, Matthew issues a GetActiveBank command to confirm that his logbook is correct and the configuration currently loaded into Register Bank 0 is stored in Register Bank 2. Register Bank 1 is being held as a fail-safe configuration setting, and the new ground station configuration is to be loaded in Register Bank 3.

Having received the required settings from the ground station provider and converted them to Elysium register values, Matthew uses the SetBlock command to upload his register settings to Bank 3 *en masse*. Because this command is going over the radio link, he sets the Reply Requested bit to get confirmation that the command was received, but does not use the CRC bit as other protocol layers make use of CRC and FEC protections to ensure that if the data does arrive, it arrives intact.

He needs to send the configuration in three blocks - one to the core firmware starting at Address 0x00, one to the Data Link Layer subsystem (in this case, SDLP) starting at 0xC0, and one to the Network Layer subsystem (in this example, Space Packet Protocol) starting at 0x80 - otherwise his SetBlock command will fail due to attempting to write to invalid register addresses. Each of these commands sends a confirmation message indicating success.

Register Bank 3 is now programmed with the appropriate settings for the new ground station. At this point, Matthew could issue the commands to execute the switch. However, his boss has been harping lately on how valuable this CubeSat is to the company and the need for constant vigilance when making significant operational changes. So Matt issues the GetBlock command three times to read out the values of the three sections of Register Bank 3 and compare them to his hard copy on the ground. As expected, everything matches up, so he sends the ReloadConfig command to load the values in Register Bank 3 into Register Bank 0 as the active configuration.

The new configuration values take effect and the whole team watches with excitement as data returns over their new ground station link! Again mindful of his boss' paranoia, Matthew sends the GetBlock command four more times, reading each of the sections of Register Bank 0 and confirming that Bank 0 now contains the correct values.

6.2 Managing On Board Radio Telemetry

Geoffrey, a systems engineer at TelemCo in charge of system-level testing for a 6U CubeSat with an X-band transmitter for payload data downlink and the Elysium for TT&C, has some concerns about the thermal load presented by the Elysium's full $5\,\mathrm{W}$ of output power. The heat sink his mechanical and thermal engineers cooked up seems a little bit flimsy to Geoff. He decides to do some in-system testing.

SSHing into the flight computer over the GSE link, he opens an interactive shell and begins setting up for his test. He issues the ChannelSub command to subscribe to the Channels for PA temperature (Power Amplifier Temperature), heat sink temperature (Heat Sink Temperature), and microcontroller temperature (Microcontroller Temperature) at a rate of once per second. He also sets the error reporting mask to include INFO using the SetErr command so that he doesn't need to increase the priority of all the relevant errors individually. Because these commands are being transmitted over the on-board UART connection, it seems to Geoffrey that commands are likely to arrive in some form, however because there is no error correction on the on-board link, they may be corrupted when they do arrive. Because of these circumstances, he decides to use the FCS function but not to use the Reply Address fucntion.

He then fires up TelemCo's standard Bit Error Rate test for the Elysium, which transmits a PN15 sequence using 4 Transfer Frames and then pauses for 5 seconds before repeating the transmission. He watches the telemetry and ses the PA temperature climb from $22\,^{\circ}\mathrm{C}$ to $43\,^{\circ}\mathrm{C}$. The heat sink temperature stabilizes around $32\,^{\circ}\mathrm{C}$, while the board temperature only increases to a paltry $24\,^{\circ}\mathrm{C}$. No warnings or errors are reported. So far, so good.

Geoff then runs the Stress Test program, which is the same as the BER test but without the pause. This time the PA temperature reaches $62\,^{\circ}\mathrm{C}$ before leveling off. Because TelemCo has set the value of the PATempWARN register to be $60\,^{\circ}\mathrm{C}$, a WARNING message is generated. The heat sink temperature hovers around $50\,^{\circ}\mathrm{C}$, and the board temperature reaches $31\,^{\circ}\mathrm{C}$.

This is the highest power and duty cycle the Elysium is capable of generating, so the test would seem to be a success. However, Geoff isn't convinced. He thinks about it for a while, and then realizes that since the X-band transmitter's heat sink is attached to the same structural panel as the Elysium heat sink, the two systems are coupled. He repeats his test with the X-band transmitter also running at its maximum duty cycle. This time the PA temperature climbs to $75\,^{\circ}\mathrm{C}$ before an ERROR message is generated and the Elysium powers off. Geoff checks his configuration and, sure enugh, the PATempERR register is set to $75\,^{\circ}\mathrm{C}$. It seems that when both transmitters are running at the same time, the thermal situation can get out of hand after all.

Fortunately, this is an easy fix for Geoff, who writes "The TT&C radio and the payload downlink radio shall not be operated in such a way as to produce simultaneous downlink events" into the operations manual and heads off to happy hour at the pub down the street.

6.3 Ground Fault

Jake, a **very** junior engineer at Cheap Labour, Inc. (CLI for short), is tasked with configuring the Elysium radios for the flight of CLI's constellation of 5 1U CubeSats. He does this using the only command he is allowed to use - the SetRegs command.

Just before launch, the FCC sends an amendment to CLI's license for their new constellation - instead of operating their uplink at $449.445\,\mathrm{MHz}$, they will now operate at $450.085\,\mathrm{MHz}$. Jake dutifully updates the value in the Register Bank that CLI has chosen to use for their primary configuration - Register Bank 1. Unfortunately, he fat-fingers the Data value and enters $450.850\,\mathrm{MHz}$ instead. He doesn't test his change because what could possibly go wrong with such a tiny update? The satellites ship on schedule.

Launch day is very exciting in Mission Operations at CLI. Everyone gathers around the big monitor for the first contact with an on-orbit satellite... and nothing happens. There's no sign of any response from the first

CubeSat. Or the second. Or the third. They even check their older operating frequency in case Jake forgot to apply the update, but there's no reply there either.

Christopher, Chief of Operations at CLI, is getting dirty looks from his bosses - the company is counting on this constellation and Somebody had better Do Something. Chris checks a few values in the config log book. Everything looks normal - there's no reason they shouldn't be able to hear from the satellites. Fortunately, the Ground Comm Lost Fault Response handler is enabled, with the timer set to 12 hours (these satellites are in an equatorial orbit). Even more fortunately, the fallback Register Bank is set to Bank 4, which was held as a backup and still contains the old operating frequency of $449.445\,\mathrm{MHz}$. Chris decides to reassure his bosses that the problem will be solved in 12 hours, and gets ready to work a night shift.

12 hours later, the radios report Ground Fault errors to their onboard processors and load their Bank 4 configurations into Bank 0. Chris picks up the response signals on the $449.445\,\mathrm{MHz}$ band and is able to re-establish contact. Once he can talk to the satellites again, he discovers Jake's error and schedules a meeting with the junior engineer to discuss the important concept of Test As You Fly.

6.4 Over-The-Air Firmware Updates

A brand new startup, Scrappy LLC, launches a 2U CubeSat as a Minimum Viable Product demonstration of their new gamma-ray Earth imaging sensor used to detect uranium and thorium deposits for mining companies. Being first time spacecraft engineers, they design their own ground station using four coathangers and a 5$ hacked-up TV dongle. Most importantly, they write their very own data link layer and install it on their Elysium radio as custom firmware, assuming they don't need the features of the "heavyweight" Space Data Link Protocol implementation.

Scrappy immediately has fantastic product-market fit. The CEO and CFO go out on Sand Hill Road to look for investors. In the meantime, the engineers are asked to look into ways to get 10x the current data downlink without any new hardware - they are assured this will really help with the terms the CEO can get from investors. So the Chief Engineer shops around and finds an up-and-coming new ground station aggregator that is offering reasonable prices. There's only one catch - the aggregator only supports the CCSDS data link layer protocols.

"No problem!", says the Chief Engineer. "Thank goodness we decided to go with the Elysium! I knew that OTA update functionality would come in handy!" He very carefully reviews the documentation on the UploadFW and InstallFW commands, then sets to work.

First, he re-compiles his firmware to use the SDLP Data Link Layer using the development virtual machine that Adamant provided Scrappy with way back when they bought their first radio. Next, he executes a careful and thorough test campaign on the ground to verify that his firmware is configured for the correct frequency bands, output power, and so forth. Then he heads over to his FlatSat testbed and prepares his plugs-out test plan. After verifying that none of the FlatSat telemetry currently stored on the Elysium is critical, he executes the UploadFW command over and over, making sure to use the FCS functionality to ensure successful transmission. Having uploaded the entire firmware image, he remembers to send the VerifyFW command with the CRC of the firmware image he generated on the ground. Oh no! Looks like he made a mistake writing his upload script and accidentally uploaded values in little-endian order in a few places! He issues the CancelFW command, fixes the bugs, and re-runs the upload and verify process. Finding no errors this time, he takes a deep breath and sends the InstallFW command. Finally, he hooks up his CCSDS-compatible receiver that he borrowed from a friend at a much larger company. Success! Everything works. Now he's ready to update the asset on orbit, make the company tons of money, cash in his stock options, and live on a beach in Aruba for the next 30 years.

6.5 Telemetry Downlink

Charles, a University student operating his lab's 1.5U CubeSat, has been seeing periodic resets in the telemetry coming down for the last several days, but isn't entirely sure why. He sets out to find out.

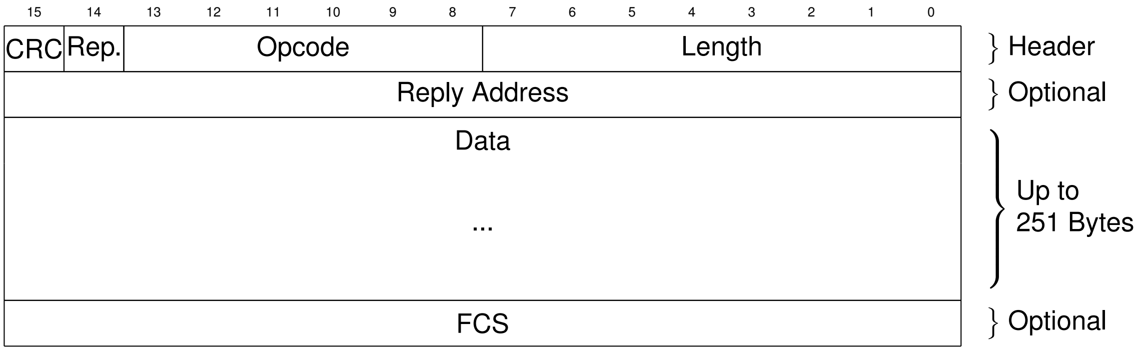
The logs show that the resets always occur off the coast of Hawaii, and that they are caused by the battery fault monitor, but battery telemetry before and after the fault seems normal. Chuck uploads commands to the battery monitor MCU to increase the frequency with which it uses the StoreTelem command to log voltage and current to the Elysium's memory from every 5 minutes to every 30 seconds. He also instructs the imager controller to increase its reporting level from WARNING to INFO. After the next reset, he sends a series of GetTelem commands, focusing on the times the CubeSat is near Hawaii to conserve precious downlink bandwidth.

After decoding the telemetry packets using the University's data dictionary, the problem is obvious. The CubeSat has been tasked to observe a pod of whales which recently migrated near one of the primary downlink sites, causing the imager and the radio to power on at the same time. The increased current draw caused the observed voltage at the battery monitor to dip, triggering the fault. Chuck escalates the problem to his thesis advisor, happy that it isn't his job to decide whether downlink or imaging is more important.

7 Commands

7.1 Command Packet Format

Packets which are sent to the Elysium radio firmware itself, rather than through the radio, have the following format:



7.1.1 CRC

The one-bit CRC flag indicates the presence or absence of the FCS packet error detection field. If the CRC flag is 1, the 16-bit CRC of the packet will be calculated as described in Section 7.1.7 and compared to the FCS field at the end of the packet. If they do not match, the packet will be discarded and an error may be reported. If the CRC flag is 0, the FCS field will be assumed not to be present.

7.1.2 Reply Requested

The one-bit Reply Requested (Rep.) flag indicates whether a confirmation packet should be sent to confirm receipt of the command. If the Rep. flag is 1, a confirmation packet will be sent. If the Rep. flag is 0, no confirmation packet will be sent. See Section 7.2 for details of the confirmation packet format.

7.1.3 Opcode

The six-bit Opcode field contains the opcode of the command which is being sent. See Section 7.3 for details of the commands and their opcodes.

The opcode of the command determines how the Data field is interpreted.

If a packet is received with an invalid Opcode, the packet is discarded and an error may be reported.

Command opcodes are drawn from the same 8-bit space as Channel, Error, and Event IDs. The six-bit Opcode field of Command packets carries only the lower 6 bits of the full Command ID. The upper 2 bits of all Command IDs are 00.

7.1.4 Length

The eight-bit Length field contains the length of the packet not including the header. For example, in a packet containing the two-byte header field, 124 bytes of data, a Reply Address, and the two-byte FCS field, the Length field should contain 128 (124 + 2 + 2), or 0x80. A zero-byte packet (such as a Reset command with no FCS) would have a Length field of 0.

7.1.5 Reply Address

The sixteen-bit Reply Address field contains the address to reply to whenever the Reply Requested flag is set. The Networking Layer subsystem will route the reply packet to the requester based on this value. Not all Networking Layer subsystems support addressing - in such cases, the reply packet will be sent according to the rules of the Networking Layer in use, and the value 0xFFFF should be placed in this field.

Table 8: FCS Algorithm Parameters

Parameter	*Value*
Polynomial	$x^{16} + x^{12} + x^5 + 1$
Polynomial (Hex)	0x1021
Reflect bits prior to processing (LSB first)?	Yes
Reflect bits prior to final XOR?	Yes
Initial Value	0xFFFF
Value of final XOR	0xFFFF

7.1.6 Data

The variable length Data field contains the data required to carry out the command. The interpretation of the field depends on the command.

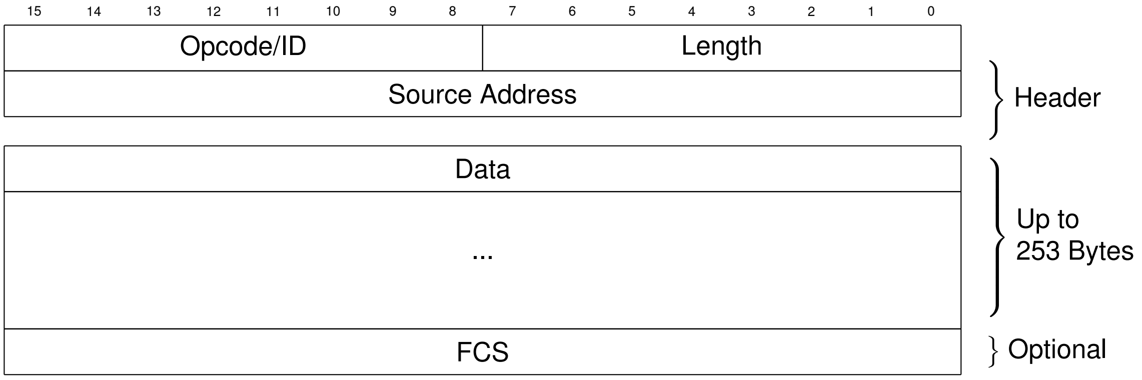
The Data field may be any length from 0 to 251 bytes, inclusive. Note that the absence of the Reply Address or FCS fields does not permit the Data field to occupy more than 251 bytes. If the Length field does not match any value expected for a command with the given Opcode, the packet is discarded and an error may be reported.

7.1.7 FCS

The FCS field contains a 16-bit CRC for the rest of the packet, including the header, whenever the CRC flag is set. The CRC used is as specified in the X.25 specification from the ITU, described as the Frame Check Sequence in section 2.2.7.4. This CRC algorithm's parameters are listed in Table 8.

7.2 Reply Packet Format

Packets which are sent by the Elysium as replies to command packets with the Reply Requested bit set have the following format:



7.2.1 Opcode/ID

The eight-bit Opcode/ID field contains the opcode of the command which is being replied to or the ID of the Channel, Error, or Event which is reporting data. See Section 7.3 for details of the commands and their opcodes. See Section 9 for details of the on board telemetry Channels and their IDs. See Section 10 for details of the available error reports, and Section 11 for details of the available events.

The opcode or ID of the reply determines how the Data field is interpreted.

7.2.2 Length

The eight-bit Length field contains the length of the packet not including the header. For example, in a packet containing the four-byte header field, 124 bytes of data, and the two-byte FCS field, the Length field should contain 126 (124 + 2), or 0x7E.

7.2.3 Source Address

The sixteen-bit Source Address field contains the address of the Elysium radio itself, as stored in the SrcAddr register. This is used to differentiate between multiple Elysium radios present within a system. It is converted to a full Network Address by the Network Layer in the same way as the Reply addresses of command packets are derived.

7.2.4 Data

The variable length Data field contains the data value of the response.

The interpretation of the field depends on the value of the Opcode/ID field. The Reply messages sent by each Command are specified in the **Response** section of the command description. The Reply messages sent to report data from each Channel are constructed from Channel data values described in the **Format** section of the Channel description.

The Data field may be any length from 0 to 253 bytes, inclusive. Note that the absence of the FCS field does not permit the Data field to occupy more than 253 bytes.

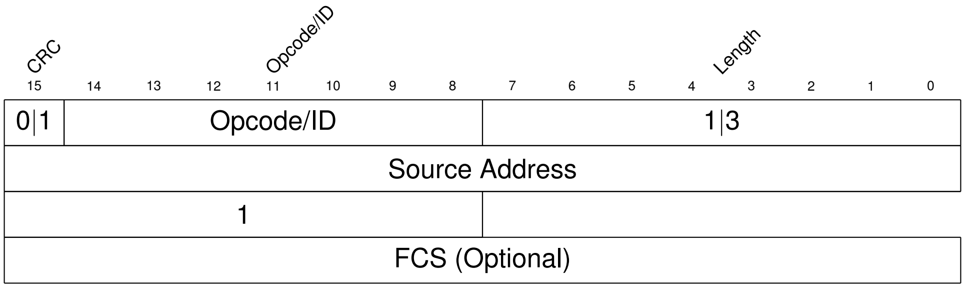
7.2.5 FCS

The FCS field contains a 16-bit CRC for the rest of the packet, including the header. The FCS will be included on all replies to commands which set the CRC bit. No Channel, Event, or Error report will contain the FCS.

7.2.6 SUCCESS and FAILURE

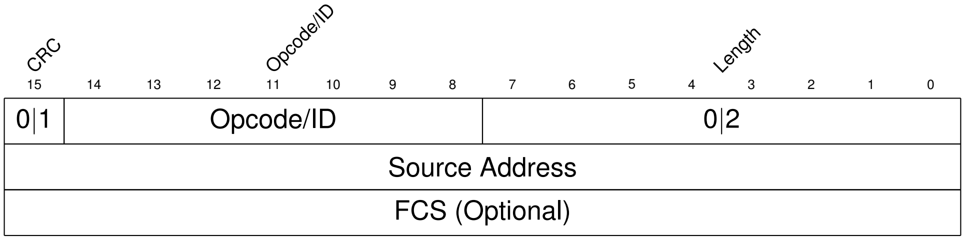
Many commands return SUCCESS or FAILURE rather than a specific data value. These messages are common enough that they are specified here rather than individually for each command.

SUCCESS:



The SUCCESS reply packet is defined as a reply packet with a single Data byte with the value 1 (or 0x01). The FCS is optional as usual.

FAILURE:



The FAILURE reply packet is defined as a reply packet with no payload (payload length of 0). The FCS is optional as usual.

7.3 List of Commands

All Elysium commands have a full name, a short-form name (this is used in the code), and an Opcode. A list of commands for the primary Elysium firmware appears in Table 9.

Table 9: Primary Firmware Commands

Opcode	*Short Form*	*Command*
0x00	Reset	Reset
0x01	Get GPO State	GetGPO
0x02	Set GPO State	SetGPO
0x03	Get Active Register Bank	GetActiveBank
0x04	Get Registers	GetRegs
0x05	Set Registers	SetRegs
0x06	Get Block	GetBlock
0x07	Set Block	SetBlock
0x08	Get Transmit Frequency	GetTXFreq
0x09	Set Transmit Frequency	SetTXFreq
0x0A	Get Receive Frequency	GetRXFreq
0x0B	Set Receive Frequency	SetRXFreq
0x0C	Get Transmit Bitrate	GetTXRate
0x0D	Set Transmit Bitrate	SetTXRate
0x0E	Get Receive Bitrate	GetRXRate
0x0F	Set Receive Bitrate	SetRXRate
0x10	Get Transmit Deviation	GetTXDev
0x11	Set Transmit Deviation	SetTXDev
0x12	Get Receive Deviation	GetRXDev
0x13	Set Receive Deviation	SetRXDev
0x14	Get Output Power	GetTXPow
0x15	Set Output Power	SetTXPow
0x16	Get UART Baud Rate	GetBaud
0x17	Set UART Baud Rate	SetBaud
0x18	Reload Configuration	ReloadConfig
0x19	Subscribe to Channels	ChannelSub
0x1A	Unsubscribe from Channels	ChannelUnsub
0x1B	Log Channels	LogChan
0x1C	Stop Logging Channels	UnlogChan
0x1D	Get Channel Value	GetChan
0x1E	Reset Channels	ResetChan
0x1F	Subscribe to Events	EventSub
0x20	Unsubscribe from Events	EventUnsub
0x21	Log Events	LogEvent
0x22	Stop Logging Events	UnlogEvent
0x23	Reset Events	ResetEvent
0x24	Set Mission Time	SetTime
0x25	Get Mission Time	GetTime
0x26	Get Error Reporting Mask	GetErr
0x27	Set Error Reporting Mask	SetErr
0x28	Get Error Logging Mask	GetLog

Table 9: Primary Firmware Commands

Opcode	*Short Form*	*Command*
0x29	Set Error Logging Mask	SetLog
0x2A	Upload New Firmware	UploadFW
0x2B	Verify New Firmware	VerifyFW
0x2C	Cancel New Firmware	CancelFW
0x2D	Install New Firmware	InstallFW
0x2E	Store Telemetry	StoreTelem
0x2F	Retrieve Telemetry	GetTelem

7.3.1 Reset

Opcode: 0x00

Arguments: None.

Description: The Reset command initiates a software reset of the Elysium radio. This is indistinguishable from a hardware reset commanded using the $\overline{\text{RESET}}$ line.

Format:

CRC	Rep.	Opcode	Length
15	14	13 12 11 10 9 8	7 6 5 4 3 2 1 0
0	0	0x00	0

Response: N/A

Notes: The Reply Requested bit is ignored by a Reset command. If the CRC bit is set, the FCS field is assumed to occupy the last 2 bytes of the packet as indicated by the Length field. The value of the Reply Address field, if any, is used to calculate the FCS but is otherwise ignored.

7.3.2 Get GPO State

Opcode: 0x01

Arguments: None.

Description: The Get GPO State command retrieves the current state of the Elysium's general purpose output (GPO) line, called GPO in Table 1.

Format:

CRC	Rep.			Opcode							Length				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0x01						2							
Reply Address															

Response: Returns the current state (0/LOW or 1/HIGH) of the GPO line.

			Opcode/ID								Length				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x01								1|3							
Source Address															
State															
FCS (Optional)															

Notes: The Reply Requested bit is required for the Get GPO State command to do anything useful.

7.3.3 Set GPO State

Opcode: 0x02

Arguments: {State}

Description: The Set GPO statecommand sets the current state of the Elysium's general purpose output (GPO) line, called GPO in Table 1.

Format:

CRC	Rep.			Opcode							Length				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0x02						1							
0-1															

Response: SUCCESS after the change has been made, or FAILURE if the requested value is out of bounds.

Notes: If the requested GPO state is invalid, the command is ignored and an error may be generated.

7.3.4 Get Active Register Bank

Opcode: 0x03

Arguments: None.

Description: The Get Active Register Bank command reports the index of the register bank from which the currently active configuration was loaded.

Format:

CRC	Rep.			Opcode							Length				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0x03						2							
Reply Address															

Response: Returns the index (1-4) of the most recently loaded register bank.

			Opcode/ID								Length				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x03								1|3							
Source Address															
Index								■							
FCS (Optional)															

Notes: The Reply Requested bit is required for the Get Active Register Bank command to do anything useful.

7.3.5 Get Registers

Opcode: 0x04

Arguments: {Bank Index, {Address} x *n*}

Description: The Get Registers command retrieves stored register values from a given Register Bank. Its data argument is a Bank Index, specifying the Register Bank to read from, followed by a series of Addresses, with each Address specifying the register whose value is to be read.

Format:

<table>
<tr><th>CRC</th><th>Rep.</th><th colspan="6">Opcode</th><th colspan="8">Length</th><th></th></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td>0</td><td>1</td><td colspan="6">0x04</td><td colspan="8"><i>n+3</i></td><td></td></tr>
<tr><td colspan="16">Reply Address</td><td></td></tr>
<tr><td colspan="8">Bank Index</td><td colspan="8">■</td><td></td></tr>
<tr><td colspan="8">Address 0</td><td colspan="8">Address 1</td><td rowspan="4">$\frac{n}{2}$</td></tr>
<tr><td colspan="8">Address 2</td><td colspan="8">Address 3</td></tr>
<tr><td colspan="8">...</td><td colspan="8">...</td></tr>
<tr><td colspan="8">Address <i>n-2</i></td><td colspan="8">Address <i>n-1</i></td></tr>
</table>

Response: Returns the values stored in the registers located at each of the listed Addresses in the indicated Register Bank, as shown below, or FAILURE if a requested register does not exist or the Bank Index is invalid.

<table>
<tr><th colspan="8">Opcode/ID</th><th colspan="8">Length</th><th></th></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td colspan="8">0x04</td><td colspan="8"><i>n</i>|<i>n+2</i></td><td></td></tr>
<tr><td colspan="16">Source Address</td><td></td></tr>
<tr><td colspan="8">Value 0</td><td colspan="8">Value 1</td><td rowspan="4">$\frac{n}{2}$</td></tr>
<tr><td colspan="8">Value 2</td><td colspan="8">Value 3</td></tr>
<tr><td colspan="8">...</td><td colspan="8">...</td></tr>
<tr><td colspan="8">Value <i>n-2</i></td><td colspan="8">Value <i>n-1</i></td></tr>
<tr><td colspan="16">FCS (Optional)</td><td></td></tr>
</table>

Notes: The Reply Requested bit is required for the Get Registers command to do anything useful.

If a requsted register does not exist or the Bank Index is invalid, the entire command will be discarded and an error may be generated.

The Get Registers command is free to access Register Bank 0, including the special registers present only in this bank.

If you need to retrieve the value of a large number of consecutive registers, you may wish to use the Get Block command.

7.3.6 Set Registers

Opcode: 0x05

Arguments: {Bank Index, {Address, Value} x *n*}

Description: The Set Registers command modifies stored register values from a given Register Bank. Its data argument is a Bank Index followed by a series of Address/Value pairs, with each Address specifying the register to be modified and each Value specifying the new value the register should be set to.

Format:

<table>
<tr><th>CRC</th><th>Rep.</th><th></th><th></th><th></th><th>Opcode</th><th></th><th></th><th></th><th></th><th></th><th></th><th>Length</th><th></th><th></th><th></th><th></th></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th><th></th></tr>
<tr><td>0</td><td>0</td><td colspan="6">0x05</td><td colspan="8"><i>2n+1</i></td><td></td></tr>
<tr><td colspan="8">Bank Index</td><td colspan="8">■</td><td></td></tr>
<tr><td colspan="8">Address 0</td><td colspan="8">Value 0</td><td rowspan="4"><i>n</i></td></tr>
<tr><td colspan="8">Address 1</td><td colspan="8">Value 1</td></tr>
<tr><td colspan="8">...</td><td colspan="8">...</td></tr>
<tr><td colspan="8">Address <i>n-1</i></td><td colspan="8">Value <i>n-1</i></td></tr>
</table>

Response: SUCCESS after all requested registers have been changed, or FAILURE if a requested register does not exist or the Bank Index is invalid.

Notes: If a requsted register does not exist or the Bank Index is invalid, the entire command will be discarded and an error may be generated. No register values will be changed.

The Set Registers command cannot access Register Bank 0 or the special registers present only in this bank. See the Reload Configuration command instead.

If you need to set the values of a large number of consecutive registers, you may wish to use the Set Block command.

7.3.7 Get Block

Opcode: 0x06

Arguments: {Bank Index, Address, Count}

Description: The Get Block command retrieves stored register values from consecutive registers of a given Register Bank. Its data argument consists of a Bank Index specifying the Register bank to read from, a single Address specifying the start of the block to read, and a Count argument specifying the size of the block.

Format:

CRC	Rep.				Opcode								Length		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0x06						5							
Reply Address															
Bank Index								Address							
Count								■							

Response: Returns the value of the block of registers, starting with the register at Address, or returns FAILURE if the block runs outside of the useful address space or the Bank Index is invalid.

				Opcode/ID								Length				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0x06								*n*|*n+2*								
Source Address																
Value 0								Value 1								$\frac{n}{2}$
Value 2								Value 3								
...								...								
Value n-2								Value n-1								
FCS (Optional)																

Notes: The Reply Requested bit is required for the Get Block command to do anything useful.

The Get Block command is free to access Register Bank 0, including the special registers present only in this bank.

We elect not to return any information if there is ambiguity in the request, on the theory that it is better not to act on potentially incorrect information.

7.3.8 Set Block

Opcode: 0x07

Arguments: {Bank Index, Address, {Value} x *n*}

Description: The Set Block command modifies stored register values in consecutive registers of a given Register Bank. Its data argument consists of a Bank Index specifying the Register Bank to modify, a single Address specifying the start of the block to modify, and a number of Values which are to be written to the block.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td colspan="6">Opcode</td><td colspan="8">Length</td><td></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td>0</td><td>0</td><td colspan="6">0x07</td><td colspan="8"><i>n</i>+2</td><td></td></tr>
<tr><td colspan="8">Bank Index</td><td colspan="8">Address</td><td></td></tr>
<tr><td colspan="8">Value 0</td><td colspan="8">Value 1</td><td rowspan="4">$\frac{n}{2}$</td></tr>
<tr><td colspan="8">Value 2</td><td colspan="8">Value 3</td></tr>
<tr><td colspan="8">...</td><td colspan="8">...</td></tr>
<tr><td colspan="8">Value <i>n-2</i></td><td colspan="8">Value <i>n-1</i></td></tr>
</table>

Response: SUCCESS after all requested registers have been changed, or FAILURE if the block runs outside of the useful address space or the Bank Index is invalid.

Notes: If the block runs outside of the useful address space or the Bank Index is invalid, the entire command will be discarded and an error may be generated. No register values will be changed.

The Set Block command cannot access Register Bank 0 or the special registers present only in this bank. See the Reload Configuration command instead.

7.3.9 Get Transmit Frequency

Opcode: 0x08

Arguments: None

Description: The Get Transmit Frequency command returns the current frequency to which the Transmit chain is tuned.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td colspan="6">Opcode</td><td colspan="8">Length</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td colspan="6">0x08</td><td colspan="8">2</td></tr>
<tr><td colspan="16">Reply Address</td></tr>
</table>

Response: Returns the value of the current transmit frequency as a big-endian 32-bit unsigned integer, as shown below.

<table>
<tr><td></td><td></td><td></td><td></td><td>Opcode/ID</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>Length</td><td></td><td></td><td></td><td></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td colspan="8">0x08</td><td colspan="8">4|6</td><td></td></tr>
<tr><td colspan="16">Source Address</td><td></td></tr>
<tr><td colspan="8">Bits 31..24</td><td colspan="8">Bits 23..16</td><td rowspan="2">Frequency (Hz)</td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td></tr>
<tr><td colspan="16">FCS (Optional)</td><td></td></tr>
</table>

Notes: The Reply Requested bit is required for the Get Transmit Frequency command to do anything useful.

7.3.10 Set Transmit Frequency

Opcode: 0x09

Arguments: {Frequency [u32]}

Description: The Set Transmit Frequency command sets the current frequency to which the Transmit chain is tuned. If the downlink is active when this command is received, the change will not take effect until after the current downlink cycle is completed.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td></td><td></td><td></td><td>Opcode</td><td></td><td></td><td></td><td></td><td></td><td></td><td>Length</td><td></td><td></td><td></td><td></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td>0</td><td>0</td><td colspan="6">0x09</td><td colspan="8">4</td><td></td></tr>
<tr><td colspan="8">Bits 31..24</td><td colspan="8">Bits 23..16</td><td rowspan="2">Frequency (Hz)</td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td></tr>
</table>

Response: SUCCESS after the frequency has actually been changed, or FAILURE if the frequency range is outside of the configured band.

Notes: The allowable frequency band for transmission is configured at compile time (before delivery) to match the hardware configuration. If custom firmware has been compiled or updates have been performed, be sure that the frequency ranges match.

7.3.11 Get Receive Frequency

Opcode: 0x0A

Arguments: None

Description: The Get Receive Frequency command returns the current frequency to which the Receive chain is tuned.

Format:

CRC	Rep.	Opcode						Length							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0x0A						2							
Reply Address															

Response: Returns the value of the current receive frequency as a big-endian 32-bit unsigned integer, as shown below.

Opcode/ID								Length								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0x0A								4|6								
Source Address																
Bits 31..24								Bits 23..16								Frequency (Hz)
Bits 15..8								Bits 7..0								
FCS (Optional)																

Notes: The Reply Requested bit is required for the Get Receive Frequency command to do anything useful.

7.3.12 Set Receive Frequency

Opcode: 0x0B

Arguments: {Frequency [u32]}

Description: The Set Receive Frequency command sets the current frequency to which the Receive chain is tuned. This command takes immediate effect.

Format:

CRC	Rep.	Opcode						Length								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0x0B						4								
Bits 31..24								Bits 23..16								Frequency (Hz)
Bits 15..8								Bits 7..0								

Response: SUCCESS after the frequency has actually been changed, or FAILURE if the frequency range is outside of the configured band.

Notes: The allowable frequency band for reception is configured at compile time (before delivery) to match the hardware configuration. If custom firmware has been compiled or updates have been performed, be sure that the frequency ranges match.

7.3.13 Get Transmit Bitrate

Opcode: 0x0C

Arguments: None

Description: The Get Transmit Bitrate command returns the current bitrate at which data is being transmitted.

Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC	Rep.	Opcode						Length							
0	1	0x0C						2							
Reply Address															

Response: Returns the value of the current transmit bitrate as a big-endian 32-bit unsigned integer, as shown below.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Opcode/ID								Length								
0x0C								4|6								
Source Address																
Bits 31..24								Bits 23..16								Bitrate (bps)
Bits 15..8								Bits 7..0								
FCS (Optional)																

Notes: The Reply Requested bit is required for the Get Transmit Bitrate command to do anything useful.

Note that this command gets the *bit*rate, prior to any channel coding. This means that the actual *data*rate may be lower than this value.

7.3.14 Set Transmit Bitrate

Opcode: 0x0D

Arguments: {Bitrate [u32]}

Description: The Set Transmit Bitrate command sets the current bitrate at which data is being transmitted. This command takes immediate effect.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td colspan="6">Opcode</td><td colspan="8">Length</td><td></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td>0</td><td>0</td><td colspan="6">0x0D</td><td colspan="8">4</td><td></td></tr>
<tr><td colspan="8">Bits 31..24</td><td colspan="8">Bits 23..16</td><td rowspan="2">Bitrate (bps)</td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td></tr>
</table>

Response: SUCCESS after the bitrate has actually been changed, or FAILURE if the bitrate is outside of the supported range.

Notes: Note that this command sets the *bit*rate, prior to any channel coding. This means that the actual *data*rate may be lower than this value.

7.3.15 Get Receive Bitrate

Opcode: 0x0E

Arguments: None

Description: The Get Receive Bitrate command returns the current bitrate at which data is being received.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td colspan="6">Opcode</td><td colspan="8">Length</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td colspan="6">0x0E</td><td colspan="8">2</td></tr>
<tr><td colspan="16">Reply Address</td></tr>
</table>

Response: Returns the value of the current receive bitrate as a big-endian 32-bit unsigned integer, as shown below.

<table>
<tr><td colspan="8">Opcode/ID</td><td colspan="8">Length</td><td></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td colspan="8">0x0E</td><td colspan="8">4|6</td><td></td></tr>
<tr><td colspan="16">Source Address</td><td></td></tr>
<tr><td colspan="8">Bits 31..24</td><td colspan="8">Bits 23..16</td><td rowspan="2">Bitrate (bps)</td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td></tr>
<tr><td colspan="16">FCS (Optional)</td><td></td></tr>
</table>

Notes: The Reply Requested bit is required for the Get Receive Bitrate command to do anything useful.

Note that this command gets the *bit*rate, prior to any channel coding. This means that the actual *data*rate may be lower than this value.

7.3.16 Set Receive Bitrate

Opcode: 0x0F

Arguments: {Bitrate [u32]}

Description: The Set Receive Bitrate command sets the current bitrate at which data is being received. This command takes immediate effect.

Format:

CRC	Rep.	Opcode	Length	
15	14	13 12 11 10 9 8	7 6 5 4 3 2 1 0	
0	0	0x0F	4	
Bits 31..24			Bits 23..16	Bitrate (bps)
Bits 15..8			Bits 7..0	

Response: SUCCESS after the bitrate has actually been changed, or FAILURE if the bitrate is outside of the supported range.

Notes: Note that this command sets the *bit*rate, prior to any channel coding. This means that the actual *data*rate may be lower than this value.

7.3.17 Get Transmit Deviation

Opcode: 0x10

Arguments: None

Description: The Get Transmit Deviation command returns the current frequency deviation with which data is being transmitted.

Format:

CRC	Rep.	Opcode	Length
15	14	13 12 11 10 9 8	7 6 5 4 3 2 1 0
0	1	0x10	2
Reply Address			

Response: Returns the value of the current transmit deviation as a big-endian 32-bit unsigned integer, as shown below.

<table>
<tr><th colspan="8">Opcode/ID</th><th colspan="8">Length</th><th></th></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td colspan="8">0x10</td><td colspan="8">4|6</td><td></td></tr>
<tr><td colspan="16">Source Address</td><td></td></tr>
<tr><td colspan="8">Bits 31..24</td><td colspan="8">Bits 23..16</td><td rowspan="2">Deviation (Hz)</td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td></tr>
<tr><td colspan="16">FCS (Optional)</td><td></td></tr>
</table>

Notes: The Reply Requested bit is required for the Get Transmit Deviation command to do anything useful.

7.3.18 Set Transmit Deviation

Opcode: 0x11

Arguments: {Deviation [u32]}

Description: The Set Transmit Deviation command sets the current frequency deviation with which data is being transmitted. If the downlink is active when this command is received, the change will not take effect until after the current downlink cycle.

Format:

<table>
<tr><th>CRC</th><th>Rep.</th><th colspan="6">Opcode</th><th colspan="8">Length</th><th></th></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td>0</td><td>0</td><td colspan="6">0x11</td><td colspan="8">4</td><td></td></tr>
<tr><td colspan="8">Bits 31..24</td><td colspan="8">Bits 23..16</td><td rowspan="2">Deviation (Hz)</td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td></tr>
</table>

Response: SUCCESS after the deviation has actually been changed, or FAILURE if the deviation is outside of the supported range.

Notes: None.

7.3.19 Get Receive Deviation

Opcode: 0x12

Arguments: None

Description: The Get Receive Deviation command returns the current frequency deviation with which data is being received.

Format:

CRC	Rep.	Opcode						Length							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0x12						2							
Reply Address															

Response: Returns the value of the current receive deviation as a big-endian 32-bit unsigned integer, as shown below.

Opcode/ID								Length								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0x12								4|6								
Source Address																
Bits 31..24								Bits 23..16								Deviation (Hz)
Bits 15..8								Bits 7..0								
FCS (Optional)																

Notes: The Reply Requested bit is required for the Get Receive Deviation command to do anything useful.

7.3.20 Set Receive Deviation

Opcode: 0x13

Arguments: {Deviation [u32]}

Description: The Set Receive Deviation command sets the current frequency deviation with which data is being received. This command takes immediate effect.

Format:

CRC	Rep.	Opcode						Length								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0x13						4								
Bits 31..24								Bits 23..16								Deviation (Hz)
Bits 15..8								Bits 7..0								

Response: SUCCESS after the deviation has actually been changed, or FAILURE if the deviation is outside of the supported range.

Notes: None.

7.3.21 Get Output Power

Opcode: 0x14

Arguments: None

Description: The Get Output Power command returns the current output power with which data is being transmitted.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td colspan="6">Opcode</td><td colspan="8">Length</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td colspan="6">0x14</td><td colspan="8">2</td></tr>
<tr><td colspan="16">Reply Address</td></tr>
</table>

Response: Returns the value of the current output power in dBm, as shown below.

<table>
<tr><td colspan="8">Opcode/ID</td><td colspan="8">Length</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="8">0x14</td><td colspan="8">1|3</td></tr>
<tr><td colspan="16">Source Address</td></tr>
<tr><td colspan="8">Power</td><td colspan="8">■</td></tr>
<tr><td colspan="16">FCS (Optional)</td></tr>
</table>

Notes: The Reply Requested bit is required for the Get Output Power command to do anything useful.

7.3.22 Set Output Power

Opcode: 0x15

Arguments: {Power}

Description: The Set Output Power command sets the output power with which data is being transmitted. This command takes immediate effect.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td colspan="6">Opcode</td><td colspan="8">Length</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td colspan="6">0x15</td><td colspan="8">1</td></tr>
<tr><td colspan="8">Power</td><td colspan="8">■</td></tr>
</table>

Response: SUCCESS after the output power has actually been changed, or FAILURE if the power is outside of the supported range (10 - $37\,\mathrm{dBm}$).

Notes: Depending on the hardware configuration, the radio may or may not be able to achieve the requested output power. If the requested power is within the 10 - $37\,\mathrm{dBm}$ range but outside the achievable power for the current configuration, the output power will be set to the nearest achievable value. So, for example, if the maximum output power of a particular radio is $2\,\mathrm{W}$ and the commanded power is $37\,\mathrm{dBm}$ $(5\,\mathrm{W})$, the resulting output power will be $33\,\mathrm{dBm}$ $(2\,\mathrm{W})$.

7.3.23 Get UART Baud Rate

Opcode: 0x16

Arguments: None

Description: The Get UART Baud Rate command returns the current UART baud rate with which data is being transmitted and received.

Format:

CRC	Rep.	Opcode						Length							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0x16						2							
Reply Address															

Response: Returns the value of the current UART baud rate as a big-endian 32-bit unsigned integer, as shown below.

Opcode/ID								Length								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0x16								4|6								
Source Address																
Bits 31..24								Bits 23..16								} Baud Rate (bps)
Bits 15..8								Bits 7..0								
FCS (Optional)																

Notes: The Reply Requested bit is required for the Get UART Baud Rate command to do anything useful.

7.3.24 Set UART Baud Rate

Opcode: 0x17

Arguments: {Baud Rate [u32]}

Description: The Set Baud Rate command sets the current UART baud rate with which data is being transmitted and received. This command takes immediate effect.

Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
CRC	Rep.				Opcode							Length				
0	0	0x17						4								
Bits 31..24								Bits 23..16								Baud Rate (bps)
Bits 15..8								Bits 7..0								

Response: SUCCESS after the baud rate has actually been changed, or FAILURE if the baud rate is outside of the supported range.

Notes: None.

7.3.25 Reload Configuration

Opcode: 0x18

Arguments: {Bank Index}

Description: The Reload Configuration command causes the Elysium to load the configuration stored in the specified Register Bank into Register Bank 0.

Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC	Rep.				Opcode							Length			
0	0	0x18						1							
Bank Index															

Response: SUCCESS after the configuration has been loaded, or FAILURE if the Bank Index is invalid.

Notes: This command does not reset the Elysium, nor does it erase stored telemetry. However, it does flush any packets that may be queued for transmission over either the radio or UART links.

If the Bank Index is invalid, the entire command is discarded and an error may be generated. No register values are changed.

7.3.26 Subscribe to Channels

Opcode: 0x19

Arguments: {Interval (ms) [u32], {Channel ID} x *n*}

Description: The Subscribe to Channels command allows a user on board the spacecraft to subscribe to telemetry channels present on board the Elysium itself, including for example the Power Amplifier temperature sensor. Channel information is reported after a given Interval, specified as a big-endian 32 bit number of milliseconds. See Section 9 for more details and a list of channels.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td></td><td></td><td></td><td>Opcode</td><td></td><td></td><td></td><td></td><td></td><td></td><td>Length</td><td></td><td></td><td></td><td></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td>0</td><td>0</td><td colspan="6">0x19</td><td colspan="8"><i>n</i>+4</td><td></td></tr>
<tr><td colspan="8">Bits 31..24</td><td colspan="8">Bits 23..16</td><td rowspan="2">Interval (ms)</td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td></tr>
<tr><td colspan="8">Channel ID 0</td><td colspan="8">Channel ID 1</td><td rowspan="4">$\frac{n}{2}$</td></tr>
<tr><td colspan="8">Channel ID 2</td><td colspan="8">Channel ID 3</td></tr>
<tr><td colspan="8">...</td><td colspan="8">...</td></tr>
<tr><td colspan="8">Channel ID <i>n-2</i></td><td colspan="8">Channel ID <i>n-1</i></td></tr>
</table>

Response: SUCCESS after the subscriptions have been updated, or FAILURE if any of the Channel IDs are invalid.

If the Reply Requested bit is set and the subscription completes successfully, channel information is sent to the address specified as the Reply Address. If the Reply Requested bit is not sent, channel information is sent to the address indicated by the ChanDefaultAddr register if permitted by the network layer. If the network layer does not support addressing and the Reply Requested bit is set, the subscription is ignored and an error may be generated.

After Interval milliseconds, packets of Channel telemetry are sent out to the subscribed address (or the default address). Channel telemetry values have the formats described in Section 9. As much as possible, Channel telemetry values being reported at the same time to the same address are grouped together in one large packet. If Channel telemetry is being sent to multiple addresses, or more telemetry is being reported than will fit in a single packet, multiple packets are generated.

Notes: The term Channels is used to differentiate between Elysium telemetry (as discussed in Section 9) and spacecraft telemetry (as discussed in the Retrieve Telemetry and Store Telemetry commands).

The Interval specified in a command packet applies to all channels subscribed to by that packet. If multiple Intervals are required (for example, thermal information at 2 second intervals and packet throughput information at 4 second intervals), multiple Subscribe to Channels command packets must be sent.

The minimum Interval at which Channel data can be produced is $100\,\mathrm{ms}$. Intervals less than this value will be rounded up to it.

One or zero Reply Address may be subscribed to a particular Channel. More than one Reply Address per Channel is not supported at this time. If a Channel is already subscribed to at a different Interval or by a different Reply Address, the new values of Interval and Reply Address override the old values and an error may be generated.

7.3.27 Unsubscribe from Channels

Opcode: 0x1A

Arguments: {Channel ID} x *n*

Description: The Unsubscribe from Channels command removes a subscription to an Elysium telemetry Channel. Channel information will no longer be sent out at fixed Intervals.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td></td><td></td><td>Opcode</td><td></td><td></td><td></td><td></td><td></td><td></td><td>Length</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td>0</td><td>0</td><td colspan="6">0x1A</td><td colspan="8"><i>n</i></td><td></td></tr>
<tr><td colspan="8">Channel ID 0</td><td colspan="8">Channel ID 1</td><td rowspan="4">$\frac{n}{2}$</td></tr>
<tr><td colspan="8">Channel ID 2</td><td colspan="8">Channel ID 3</td></tr>
<tr><td colspan="8">...</td><td colspan="8">...</td></tr>
<tr><td colspan="8">Channel ID <i>n-2</i></td><td colspan="8">Channel ID <i>n-1</i></td></tr>
</table>

Response: SUCCESS after the subscriptions have been deleted, or FAILURE if any of the Channel IDs are outside of the supported range.

Notes: It is permitted to unsubscribe from channels with no active subscriptions - this will not generate a FAILURE message.

It is not permitted to unsubscribe from invalid channels. If any of the Channel IDs are invalid, the entire command is ignored and an error may be generated.

No protections are in place to ensure that only the subscribing Reply Address may unsubscribe from a given Channel. If this property is desired it must be ensured at a higher level.

7.3.28 Log Channels

Opcode: 0x1B

Arguments: {Interval (ms) [u32], {Channel ID} x *n*}

Description: The Log Channels command allows a user to log values from the telemetry channels on board the Elysium to the on board non-volatile storage for later retrieval or downlink. Channel information is logged after a given Interval, specified as a big-endian 32 bit number of milliseconds. See Section 9 for more details and a list of channels.

Format:

<table>
<tr><th>CRC</th><th>Rep.</th><th colspan="6">Opcode</th><th colspan="8">Length</th><th></th></tr>
<tr><th>15</th><th>14</th><th>13</th><th>12</th><th>11</th><th>10</th><th>9</th><th>8</th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th><th></th></tr>
<tr><td>0</td><td>0</td><td colspan="6">0x1B</td><td colspan="8"><i>n</i>+4</td><td></td></tr>
<tr><td colspan="8">Bits 31..24</td><td colspan="8">Bits 23..16</td><td rowspan="2">Interval (ms)</td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td></tr>
<tr><td colspan="8">Channel ID 0</td><td colspan="8">Channel ID 1</td><td rowspan="4">$fracn2$</td></tr>
<tr><td colspan="8">Channel ID 2</td><td colspan="8">Channel ID 3</td></tr>
<tr><td colspan="8">...</td><td colspan="8">...</td></tr>
<tr><td colspan="8">Channel ID <i>n-2</i></td><td colspan="8">Channel ID <i>n-1</i></td></tr>
</table>

Response: SUCCESS after the logging has begun, or FAILURE if any of the channel IDs are outside of the supported range.

After Interval milliseconds, packets of Channel telemetry are sent to the on board storage for logging. Channel telemetry values have the formats described in Section 9. As much as possible, Channel telemetry values being logged at the same time are grouped together into one large telemetry value. If more telemetry is being reported than will fit in a single telemetry slot, multiple values are generated.

Channel telemetry is packed into on board storage by direct concatenation. The channel ID value which is part of all reported channel telemetry is used to determine the size of each individual channel telemetry item. See Section 9 for the complete picture.

Notes: The term Channels is used to differentiate between Elysium telemetry (as discussed in Section 9) and spacecraft telemetry (as discussed in the Retrieve Telemetry and Store Telemetry commands). After being logged, Channel data becomes Telemetry.

The Interval specified in a command packet applies to all channels logged by that packet. If multiple Intervals are required (for example, thermal information at 2 second intervals and packet throughput information at 4 second intervals), multiple Log Channels command packets must be sent.

7.3.29 Stop Logging Channels

Opcode: 0x1C

Arguments: {Channel ID} x *n*

Description: The Stop Logging Channels command ends the logging of an Elysium telemetry Channel. Channel information will no longer be logged to on board storage at fixed Intervals.

Format:

CRC	Rep.	Opcode						Length								
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0x1C						*n*								
Channel ID 0								Channel ID 1								$\frac{n}{2}$
Channel ID 2								Channel ID 3								
...								...								
Channel ID *n-2*								Channel ID *n-1*								

Response: SUCCESS after the logging has stopped, or FAILURE if any of the Channel IDs are outside of the supported range.

Notes: It is permitted to stop logging channels which are not currently being logged - this will not generate a FAILURE message.

It is not permitted to stop logging invalid channels. If any of the Channel IDs are invalid, the entire command is ignored and an error may be generated.

7.3.30 Get Channel Value

Opcode: 0x1D

Arguments: {Channel ID}

Description: The Get Channel Value command retrieves the instantaneous value of an Elysium telemetry channel.

Format:

CRC	Rep.	Opcode						Length							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0x1D						3							
Reply Address															
Channel ID															

Response: Returns the current value of the channel specified by Channel ID, as specified in Section 9, or FAILURE if the Channel ID is invalid.

Notes: The Reply Requested bit is required for the Get Channel Value command to do anything useful.

If the Channel ID is invalid, the entire command is ignored and an error may be generated.

7.3.31 Reset Channels

Opcode: 0x1E

Arguments: None.

Description: The Reset Channels command resets all subscription and logging of channels.

Format:

CRC	Rep.			Opcode								Length			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0x1E						0							

Response: SUCCESS after the channels have been reset.

Notes: After executing this command, no channels are reported and no channel values are logged until the radio begins receiving new Subscribe to Channels or Log Channels commands.

7.3.32 Subscribe to Events

Opcode: 0x1F

Arguments: {Event ID} x *n*

Description: The Subscribe to Events command allows a user on board the spacecraft to subscribe to event notifications from the Elysium, including for example the Mission Time being changed. Event notifications are reported asynchronously as they occur. See Section 11 for more details and a list of events.

Format:

CRC	Rep.			Opcode								Length				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0x1F						*n*								
Event ID 0								Event ID 1								$\frac{n}{2}$
Event ID 2								Event ID 3								
...								...								
Event ID *n-2*								Event ID *n-1*								

Response: SUCCESS after the subscriptions have been updated, or FAILURE if any of the Event IDs are invalid.

If the Reply Requested bit is set and the subscription completes successfully, event notifications are sent to the address specified as the Reply Address. If the Reply Requested bit is not sent, event notifications are sent to the address indicated by the EventDefaultAddr register if permitted by the network layer. If the network layer does not support addressing and the Reply Requested bit is set, the subscription is ignored and an error may be generated.

When an event occurs, Event notification packets are sent out to the subscribed address (or the default address). Event notification packets have the formats described in Section 11. Each event notification is sent as its own packet.

Notes: One or zero Reply Address may be subscribed to a particular Event. More than one Reply Address per Event is not supported at this time. If a Event is already subscribed to by a different Reply Address, the new value of the Reply Address overrides the old value and an error may be generated.

7.3.33 Unsubscribe from Events

Opcode: 0x20

Arguments: {Event ID} x *n*

Description: The Unsubscribe from Events command removes a subscription to an Elysium event. Event notifications will no longer be sent out when events occur.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td colspan="6">Opcode</td><td colspan="8">Length</td><td></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td>0</td><td>0</td><td colspan="6">0x20</td><td colspan="8"><i>n</i></td><td></td></tr>
<tr><td colspan="8">Event ID 0</td><td colspan="8">Event ID 1</td><td rowspan="4">$\frac{n}{2}$</td></tr>
<tr><td colspan="8">Event ID 2</td><td colspan="8">Event ID 3</td></tr>
<tr><td colspan="8">...</td><td colspan="8">...</td></tr>
<tr><td colspan="8">Event ID <i>n-2</i></td><td colspan="8">Event ID <i>n-1</i></td></tr>
</table>

Response: SUCCESS after the subscriptions have been deleted, or FAILURE if any of the Event IDs are outside of the supported range.

Notes: It is permitted to unsubscribe from events with no active subscriptions - this will not generate a FAILURE message.

It is not permitted to unsubscribe from invalid events. If any of the Event IDs are invalid, the entire command is ignored and an error may be generated.

No protections are in place to ensure that only the subscribing Reply Address may unsubscribe from a given Event. If this property is desired it must be ensured at a higher level.

7.3.34 Log Events

Opcode: 0x21

Arguments: {Event ID} x *n*

Description: The Log Events command allows a user to log event notifications to the on board non-volatile storage for later retrieval or downlink. Event notifications are logged asynchronously, as they occur. See Section 11 for more details and a list of events.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td colspan="6">Opcode</td><td colspan="8">Length</td><td></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td>0</td><td>0</td><td colspan="6">0x21</td><td colspan="8"><i>n</i></td><td></td></tr>
<tr><td colspan="8">Event ID 0</td><td colspan="8">Event ID 1</td><td rowspan="4">$\left.\right\}\frac{n}{2}$</td></tr>
<tr><td colspan="8">Event ID 2</td><td colspan="8">Event ID 3</td></tr>
<tr><td colspan="8">...</td><td colspan="8">...</td></tr>
<tr><td colspan="8">Event ID <i>n-2</i></td><td colspan="8">Event ID <i>n-1</i></td></tr>
</table>

Response: SUCCESS after the logging has begun, or FAILURE if any of the Event IDs are outside of the supported range.

When an event occurs, Event notification packets are sent to the on board storage for logging. Event notification packets have the formats described in Section 11. As much as possible, event notifications being logged at the same time (within the timestamped second) are grouped together into one large telemetry value. If more telemetry is being reported than will fit in a single telemetry slot, multiple values are generated.

Event notifications are packed into on board storage by direct concatenation. All event notifications have the same size. See Section 11 for the complete picture.

Notes: None.

7.3.35 Stop Logging Events

Opcode: 0x22

Arguments: {Event ID} x *n*

Description: The Stop Logging Events command ends the logging of an Elysium event. Event notifications will no longer be logged to on board storage as events occur.

Format:

CRC 15	Rep. 14	Opcode 13 12 11 10 9 8	Length 7 6 5 4 3 2 1 0	
0	0	0x22	*n*	
Event ID 0			Event ID 1	$\frac{n}{2}$
Event ID 2			Event ID 3	
...			...	
Event ID *n-2*			Event ID *n-1*	

Response: SUCCESS after the logging has stopped, or FAILURE if any of the Event IDs are outside of the supported range.

Notes: It is permitted to stop logging events which are not currently being logged - this will not generate a FAILURE message.

It is not permitted to stop logging invalid events. If any of the Event IDs are invalid, the entire command is ignored and an error may be generated.

7.3.36 Reset Events

Opcode: 0x23

Arguments: None.

Description: The Reset Events command resets all subscription and logging of events.

Format:

CRC 15	Rep. 14	Opcode 13 12 11 10 9 8	Length 7 6 5 4 3 2 1 0
0	0	0x23	0

Response: SUCCESS after the events have been reset.

Notes: After executing this command, no events are reported or logged until the radio begins receiving new Subscribe to Events or Log Events commands.

7.3.37 Set Mission Time

Opcode: 0x24

Arguments: {Time (s) [i32]}

Description: The Set Mission Time command is used to set the current time for the Elyisum in seconds since some epoch.

Format:

CRC	Rep.			Opcode							Length					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0x24						4								
Bits 31..24								Bits 23..16								Time (s)
Bits 15..8								Bits 7..0								

Response: The Response packet for the Set Mission Time command also includes the time, as shown below. This is used to allow the Elysium to inform the rest of the spacecraft of a time update from the ground.

				Opcode/ID							Length					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0x24								4|6								
Source Address																
Bits 31..24								Bits 23..16								Time (s)
Bits 15..8								Bits 7..0								
FCS (Optional)																

Notes: Because the Set Mission Time command takes only a signed number of seconds, it can be used with regards to any epoch which is convenient to the mission requirements. Common choices include GPS time, J2000, and UNIX time, but arbitrary epochs are supported.

7.3.38 Get Mission Time

Opcode: 0x25

Arguments: None

Description: The Get Mission Time command returns the current Mission Time in seconds.

Format:

CRC	Rep.			Opcode							Length				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0x25						2							
Reply Address															

Response: Returns the current value of Mission Time, as shown below.

				Opcode/ID								Length				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0x25								4|6								
Source Address																
Bits 31..24								Bits 23..16								Time (s)
Bits 15..8								Bits 7..0								
FCS (Optional)																

Notes: The Reply Requested bit is required for the Get Mission Time command to do anything useful.

Mission Time is set using the Set Mission Time command, and is counted by seconds from that point. Mission Time persists through any reboots of the Elysium, although some error will likely be introduced.

Mission Time is accurate to within $100\frac{\mu s}{s}$. This limit is set by the frequency stability of the microcontroller's oscillator.

7.3.39 Get Error Reporting Mask

Opcode: 0x26

Arguments: None.

Description: The Get Error Reporting Mask command returns the mask describing the criticality of errors which will be reported by the Elysium.

Format:

CRC	Rep.				Opcode							Length			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0x26						2							
Reply Address															

Response: Returns the current value of the error priority mask, as shown below.

				Opcode/ID								Length			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x26								1|3							
Source Address															
Mask								■							
FCS (Optional)															

Notes: The Reply Requested bit is required for the Get Error Reporting Mask command to do anything useful.

See the Set Error Reporting Mask command for more details.

7.3.40 Set Error Reporting Mask

Opcode: 0x27

Arguments: {Priority Mask}

Description: The Set Error Reporting Mask command configures the criticality of errors which will be reported by the Elysium. There are 5 levels of criticality, represented as shown in Table 13. The Data Value of this command is a mask of priority levels to report.

Format:

CRC	Rep.				Opcode							Length			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0x27						1							
Priority Mask								■							

Response: SUCCESS after the error mask has been changed, or FAILURE if the value of *Priority Mask* is not a valid combination of the accepted values shown in Table 13.

Notes: The list of errors, their formats, and the conditions which may generate them can be found in Section 10.

The priority of some errors may be modified by the values of particular configuration registers. Such errors will make note of this fact in the **Priority Register** section of their entry in the list of errors.

Errors are edge-triggered - that is, exceeding the LNA overcurrent threshold produces an LNA Overcurrent error, but no further LNA Overcurrent errors are produced until after the LNA curren has decreased below the threshold.

All errors whose priority level is set in this mask will be reported.

In addition to being reported, errors may be logged to on-board telemetry. See the Set Error Logging Mask command for more details.

Errors are reported to the address stored in the ErrRptAddr register. This may be a broadcast address if the network layer supports it.

7.3.41 Get Error Logging Mask

Opcode: 0x28

Arguments: None.

Description: The Get Error Logging Mask command returns the mask describing the criticality of errors which will be logged to the Elysium on board storage as telemetry.

Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC	Rep.	Opcode						Length							
0	1	0x28						2							
Reply Address															

Response: Returns the current value of the error priority mask, as shown below.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Opcode/ID								Length							
0x28								1|3							
Source Address															
Mask															
FCS (Optional)															

Notes: The Reply Requested bit is required for the Get Error Logging Mask command to do anything useful.

See the Set Error Logging Mask command for more details.

7.3.42 Set Error Logging Mask

Opcode: 0x29

Arguments: {Priority Mask}

Description: The Set Error Logging Mask command configures the criticality of errors which will be logged to the Elysium on board storage as telemetry. The Data Value of this command is a mask of priority levels to be logged.

Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC	Rep.	Opcode						Length							
0	0	0x29						1							
Priority Mask															

Response: SUCCESS after the priority mask has been changed, or FAILURE if the value of *Priority Mask* is not a valid combination of the accepted values shown in Table 13.

Notes: The list of errors, their formats, and the conditions which may generate them can be found in Section 10.

The priority of some errors may be modified by the values of particular configuration registers. Such errors will make note of this fact in the **Priority Register** section of their entry in the list of errors.

Errors are edge-triggered - that is, exceeding the LNA overcurrent threshold produces an LNA Overcurrent error, but no further LNA Overcurrent errors are produced until after the LNA curren has decreased below the threshold.

All errors whose priority level is set in this mask will be logged.

In addition to being logged, errors may be reported as well. See the Set Error Reporting Mask command for more details.

7.3.43 Upload New Firmware

Opcode: 0x2A

Arguments: {Address [u32], {Data Byte} x *n*}

Description: The Upload New Firmware command uploads chunks of binary data to the New Firmware staging area.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td colspan="6">Opcode</td><td colspan="8">Length</td><td></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td>1</td><td>1</td><td colspan="6">0x2A</td><td colspan="8"><i>n</i>+8</td><td></td></tr>
<tr><td colspan="16">Reply Address</td><td></td></tr>
<tr><td colspan="8">Bits 31..24</td><td colspan="8">Bits 23..16</td><td rowspan="2">Address</td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td></tr>
<tr><td colspan="8">Data Byte 0</td><td colspan="8">Data Byte 1</td><td rowspan="4">$\frac{n}{2}$</td></tr>
<tr><td colspan="8">Data Byte 2</td><td colspan="8">Data Byte 3</td></tr>
<tr><td colspan="8">...</td><td colspan="8">...</td></tr>
<tr><td colspan="8">Data Byte <i>n-2</i></td><td colspan="8">Data Byte <i>n-1</i></td></tr>
<tr><td colspan="16">FCS</td><td></td></tr>
</table>

Response: SUCCESS after the data bytes have been written to the staging area, or FAILURE if the data block would run outside of the acceptable address range.

Notes: It is recommended to always use the FCS and Reply Address functions when uploading new firmware, especially from the ground, as a partial or corrupted firmware update could permanently damage the radio. See also the Verify New Firmware command.

The information contained in this document is sufficient to install updates provided by Adamant. More detailed information about the Firmware Upload commands is located in a confidential document giving information on the development of custom firmware and containing implementation details of the Elysium radio. This document is available free of charge to anyone willing to sign a non-disclosure agreement protecting Adamant's intellectual property. Please contact Adamant for our standard NDA or to discuss in more detail.

7.3.44 Verify New Firmware

Opcode: 0x2B

Arguments: {CRC [u16]}

Description: The Verify Firmware command is intended to verify the CRC signature of the entire New Firmware staging area.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td colspan="6">Opcode</td><td colspan="8">Length</td><td></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td>0</td><td>1</td><td colspan="6">0x2B</td><td colspan="8">4</td><td></td></tr>
<tr><td colspan="16">Reply Address</td><td></td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td><td>} CRC</td></tr>
</table>

Response: SUCCESS if the signature of the New Firmware staging area matches the uploaded CRC value, FAILURE otherwise.

Notes: This command is not intended to replace the packet-level FCS checks on the individual data chunks uploaded by the Upload New Firmware command. It is intended to verify at the firmware level that all chunks have been uploaded correctly and in the correct order. 16-bit CRC values are insufficient to reliably detect individual bit errors in a firmware image as large as $64\,\mathrm{KB}$.

The CRC value used in this command is calculated in the same way as the Frame Check Sequence, as described in Section 7.1.7

The Reply Requested bit is required for the Verify New Firmware command to do anything useful.

The information contained in this document is sufficient to install updates provided by Adamant. More detailed information about the Firmware Upload commands is located in a confidential document giving information on the development of custom firmware and containing implementation details of the Elysium radio. This document is available free of charge to anyone willing to sign a non-disclosure agreement protecting Adamant's intellectual property. Please contact Adamant for our standard NDA or to discuss in more detail.

7.3.45 Cancel New Firmware

Opcode: 0x2C

Arguments: None

Description: The Cancel New Firmware command cancels the upload of firmware initiated by the Upload New Firmware command and clears out the New Firmware staging area.

Format:

CRC	Rep.	Opcode						Length							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0x2C						0							

Response: SUCCESS after the firmware is cleared out.

Notes: This command actually writes a "cleared" value to the New Firmware staging area to prevent old firmware images from mixing with new firmware images should another firmware upload be attempted.

The information contained in this document is sufficient to install updates provided by Adamant. More detailed information about the Firmware Upload commands is located in a confidential document giving information on the development of custom firmware and containing implementation details of the Elysium radio. This document is available free of charge to anyone willing to sign a non-disclosure agreement protecting Adamant's intellectual property. Please contact Adamant for our standard NDA or to discuss in more detail.

7.3.46 Install New Firmware

Opcode: 0x2D

Arguments: None

Description: The Install New Firmware command initiates the install of firmware uploaded by the Upload New Firmware command. The Elysium resets into a small bootloader and copies over every byte of firmware from the New Firmware staging area into the main memory area. The Elysium then resets again, booting into the new firmware.

Format:

CRC	Rep.	Opcode						Length							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0x2D						0							

Response: SUCCESS after command received, before the initial reset takes place.

Notes: Because it overwrites all existing memory, this process is very likely to reset things like counters and other volatile and non-volatile storage, unless the firmware image is specifically designed to maintain their values.

A reset during this process may cause the radio to become unusable only if the area of memory corresponding to the bootloader is being updated at the time. This is a very small area of code which minimizes the risk of serious error. No firmware upgrade process is entirely safe. Exercise all due caution.

The information contained in this document is sufficient to install updates provided by Adamant. More detailed information about the Firmware Upload commands is located in a confidential document giving information on the development of custom firmware and containing implementation details of the Elysium radio. This document is available free of charge to anyone willing to sign a non-disclosure agreement protecting Adamant's intellectual property. Please contact Adamant for our standard NDA or to discuss in more detail.

7.3.47 Store Telemetry

Opcode: 0x2E

Arguments: {Data Byte} x *n*

Description: The Store Telemetry command stores opaque telemetry values to the Elysium's onboard memory for later downlink. It is intended to be used when primary flight computers may not always be powered during downlink passes.

Format:

CRC	Rep.				Opcode								Length			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
0	0	0x2E						*n*								
Data Byte 0								Data Byte 1								$\frac{n}{2}$
Data Byte 2								Data Byte 3								
...								...								
Data Byte *n-2*								Data Byte *n-1*								

Response: SUCCESS after the telemetry value has been written to onboard storage.

Notes: Telemetry values are timestamped using the current Mission Time as propagated from the last Set Mission Time command. If a Set Mission Time command is received between telemetry values, their timestamps may appear to run backwards or leap forwards according to the new Mission Time.

Telemetry values are stored in fixed-length slots occupying 255 bytes including four-byte timestamp. Each Store Telemetry command is stored in its own slot. As the maximum length of a command packet's data field is 251 bytes, no space is wasted when maximum length Store Telemetry command packets are used. When shorter Store Telemetry command packets are received, the remaining data in the slot is undefined.

256 telemetry slots exist, each with a unique index in the range [0-255]. Arriving telemetry is stored into the next slot in the sequence, wrapping around from slot 255 back to slot 0. Older telemetry is overwritten by newly arriving telemetry.

When Channel data is logged using the logging feature described in Section 9, all Channel data arriving within 1 second is logged to the same telemetry slot when possible. When more than 256 bytes of Channel data arrives within the same second, multiple slots are used. Channel values are packed into slots by simple concatenation - the unique ID of the channel is sufficient information to unpack the values later on. The same process is used to pack Errors which are logged according to the mask stored in the ErrLogLvl register or Events which are logged due to the Log Events command. Events, Errors, and Channel data may be packed into the same slot - their IDs are unique.

For details of the format of telemetry downlink, see the Retrieve Telemetry command.

7.3.48 Retrieve Telemetry

Opcode: 0x2F

Arguments: {Duration (s) [u16], (Index Start [u8], Index End [u8]), (Timestamp Start [u32], Timestamp End [u32])}

Description: The Retrieve Telemetry command is used to retrieve telemetry from the Elysium's onboard storage that has previously been stored using the Store Telemetry command or through the logging feature as described in Section 9. Telemetry is retrieved from oldest stored value to youngest for a specified Duration (in seconds) and is optionally filtered by either index or timestamp.

Format:

<table>
<tr><td>CRC</td><td>Rep.</td><td colspan="6">Opcode</td><td colspan="8">Length</td><td></td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td>0</td><td>1</td><td colspan="6">0x2F</td><td colspan="8">14</td><td></td></tr>
<tr><td colspan="16"><i>Reply Address</i></td><td></td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td><td>} Duration (s)</td></tr>
<tr><td colspan="8">Index Start</td><td colspan="8">Index End</td><td></td></tr>
<tr><td colspan="8">Bits 31..24</td><td colspan="8">Bits 23..16</td><td rowspan="2">} Timestamp Start (s)</td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td></tr>
<tr><td colspan="8">Bits 31..24</td><td colspan="8">Bits 23..16</td><td rowspan="2">} Timestamp End (s)</td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td></tr>
</table>

Response: Downlink packets constructed from telemetry slots are formatted as follows:

<table>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td></tr>
<tr><td colspan="8">Index Start</td><td colspan="8">Index End</td><td></td></tr>
<tr><td colspan="8">Bits 31..24</td><td colspan="8">Bits 23..16</td><td rowspan="2">} Timestamp (s)</td></tr>
<tr><td colspan="8">Bits 15..8</td><td colspan="8">Bits 7..0</td></tr>
<tr><td colspan="8">Index</td><td colspan="8">Data Byte 0</td><td rowspan="4">} 252 Bytes</td></tr>
<tr><td colspan="8">Data Byte 1</td><td colspan="8">Data Byte 2</td></tr>
<tr><td colspan="8">...</td><td colspan="8">...</td></tr>
<tr><td colspan="8">Data Byte 249</td><td colspan="8">Data Byte 250</td></tr>
</table>

Including the Index allows the operator to determine whether crucial telemetry is missing and request its retransmission if necessary.

Notes: The Format rendered here includes the use of both types of optional filtering. Neither the timestamp filtering nor the index filtering is required.

Telemetry packets requested by the Retrieve Telemetry command will be transmitted at a lower priority than live packets coming in over the UART link.

8 Configuration Registers

The Elysium radio provides 4 redundant banks of configuration registers, indexed as Bank 1 through Bank 4, which contain the complete configuration of the radio. These banks of registers can be modified using the SetRegs and SetBlock commands. The active configuration of the radio is stored in Register Bank 0, which is read-only. All register banks, including Bank 0, can be read using the GetRegs and GetBlock commands. The configuration from a stored Register Bank can be loaded into Bank 0 using the ReloadConfig command, while the most recently loaded Register Bank can be determined using the GetActiveBank command. An example of reconfiguration can be found in the Modifying Configuration example.

Due to the architecture of the Elysium, all register values persist through resets or reboots of the on board microcontroller. While this is generally a good thing, care must be taken to avoid a situation where it is impossible to prevent recovery from a misconfiguration.

Each register has a range of acceptable values. When this range is different than the acceptable range for the data type stored in the register, it will be noted in the Notes section. Attempts to program registers outside the acceptable range will be clipped to the nearest valid value. Depending on the current configuration, a warning message may be generated.

The configuration registers, their default values (if any), their effects and their restrictions can be found in the following sections.

Register Bank 0 contains some special, read-only registers which are not present in the other banks. These registers are described in Section 8.2.

8.1 General Registers

This section defines the general registers listed in Table 10, which exist in all 5 Register Banks (0-4) and may be modified by the Set Registers and Set Block commands.

The address range from 0x00 to 0x5F is reserved for the General Registers.

Table 10: General Registers

Address	*Name*	*Description*
0x00	TXFreq0	Transmit Frequency LSB
0x01	TXFreq1	Transmit Frequency Low Middle Byte
0x02	TXFreq2	Transmit Frequency High Middle Byte
0x03	TXFreq3	Transmit Frequency MSB

Table 10: General Registers

Address	*Name*	*Description*
0x04	RXFreq0	Receive Frequency LSB
0x05	RXFreq1	Receive Frequency Low Middle Byte
0x06	RXFreq2	Receive Frequency High Middle Byte
0x07	RXFreq3	Receive Frequency MSB
0x08	TXDev0	Transmit Deviation LSB
0x09	TXDev1	Transmit Deviation Low Middle Byte
0x0A	TXDev2	Transmit Deviation High Middle Byte
0x0B	TXDev3	Transmit Deviation MSB
0x0C	RXDev0	Receive Deviation LSB
0x0D	RXDev1	Receive Deviation Low Middle Byte
0x0E	RXDev2	Receive Deviation High Middle Byte
0x0F	RXDev3	Receive Deviation MSB
0x10	TXSync0	Transmit Sync Word LSB
0x11	TXSync1	Transmit Sync Word Low Middle Byte
0x12	TXSync2	Transmit Sync Word High Middle Byte
0x13	TXSync3	Transmit Sync Word MSB
0x14	RXSync0	Receive Sync Word LSB
0x15	RXSync1	Receive Sync Word Low Middle Byte
0x16	RXSync2	Receive Sync Word High Middle Byte
0x17	RXSync3	Receive Sync Word MSB
0x18	TXBR0	Transmit Bitrate LSB
0x19	TXBR1	Transmit Bitrate Low Middle Byte
0x1A	TXBR2	Transmit Bitrate High Middle Byte
0x1B	TXBR3	Transmit Bitrate MSB
0x1C	RXBR0	Receive Bitrate LSB
0x1D	RXBR1	Receive Bitrate Low Middle Byte
0x1E	RXBR2	Receive Bitrate High Middle Byte
0x1F	RXBR3	Receive Bitrate MSB
0x20	FilterParams	Filter Configuration Bitfields
0x21	OutputPower	Transmitter output Power
0x22	UARTBaud0	UART Baud Rate LSB
0x23	UARTBaud1	UART Baud Rate Low Middle Byte
0x24	UARTBaud2	UART Baud Rate High Middle Byte
0x25	UARTBaud3	UART Baud Rate MSB
0x26	UARTParams	UART Configuration Bitfields
0x27	FaultResponse	Fault Response Configuration Bitfields
0x28	ErrorReport0	Error Reporting Bitfields LSB
0x29	ErrorReport1	Error Reporting Bitfields MSB
0x2A	ResetErrLvl	Reset Error Priority Level
0x2B	UARTErrLvl	UART Error Priority Level
0x2C	ResetErrLvl	Subscription Overwritten Error Priority Level

Table 10: General Registers

Address	*Name*	*Description*
0x2D	NRErrLvl	Command Data Value Error Priority Level
0x2E	FCSLvl	Frame Check Sequence Error Priority Level
0x2F	LengthErrLvl	Command Packet Length Error Priority Level
0x30	OpErrLvl	Command Opcode Error Priority Level
0x31	RegErrLvl	Register Value Clipping Error Priority Level
0x32	SCCommLvl	Spacecraft Communication Loss Error Priority Level
0x33	GFLvl	Ground Communication Loss Error Priority Level
0x34	SCCommTime0	Spacecraft Comm Fault Timer LSB
0x35	SCCommTime1	Spacecraft Comm Fault Timer Low Middle Byte
0x36	SCCommTime2	Spacecraft Comm Fault Timer High Middle Byte
0x37	SCCommTime3	Spacecraft Comm Fault Timer MSB
0x38	SCCommBaud0	Spacecraft Comm Fault Baud Rate LSB
0x39	SCCommBaud1	Spacecraft Comm Fault Baud Rate Low Middle Byte
0x3A	SCCommBaud2	Spacecraft Comm Fault Baud Rate High Middle Byte
0x3B	SCCommBaud3	Spacecraft Comm Fault Baud Rate MSB
0x3C	GFTime0	Ground Comm Fault Timer LSB
0x3D	GFTime1	Ground Comm Fault Timer Low Middle Byte
0x3E	GFTime2	Ground Comm Fault Timer High Middle Byte
0x3F	GFTime3	Ground Comm Fault Timer MSB
0x40	BeaconTime0	Beacon Mode Timer LSB
0x41	BeaconTime1	Beacon Mode Timer Low Middle Byte
0x42	BeaconTime2	Beacon Mode Timer High Middle Byte
0x43	BeaconTime3	Beacon Mode Timer MSB
0x44	GFBank	Ground Comm Fault Fallback Register Bank
0x45	OTFaultTime	Overtemp Comm Fault Timer
0x46	BoardTempWARN	MCU Temperature WARNING Threshold
0x47	BoardTempERR	MCU Temperature ERROR Threshold
0x48	HSTempWARN	Heat Sink Temperature WARNING Threshold
0x49	HSTempERR	Heat Sink Temperature ERROR Threshold
0x4A	PATempWARN	Power Amplifier Temperature WARNING Threshold
0x4B	PATempERR	Power Amplifier Temperature ERROR Threshold
0x4C	ErrRptLvl	Error Reporting Priority Mask
0x4D	ErrLogLvl	Error Logging Priority Mask
0x4E	ErrRptAddr0	Error Reporting Address LSB
0x4F	ErrRptAddr1	Error Reporting Address MSB
0x50	SrcAddr0	Elysium Source Address LSB
0x51	SrcAddr1	Elysium Source Address MSB
0x52	ChanDefaultAddr0	Channel Reporting Default Address LSB
0x53	ChanDefaultAddr1	Channel Reporting Default Address MSB
0x54	EventDefaultAddr0	Event Reporting Default Address LSB
0x55	EventDefaultAddr1	Event Reporting Default Address MSB

8.1.1 TXFreq[0-3]

Address: 0x00

Data Type: uint32_t (Hz)

Description: The TXFreq register contains the transmit frequency for the Elysium as a 32-bit unsigned integer in Hz.

Diagram:

Register 8.1: TXFREQ (0x00)

TXFreq3	TXFreq2	TXFreq1	TXFreq0	
31 24	23 16	15 8	7 0	
0 0 0 1 1 0 1 1	1 1 0 0 1 1 1 1	0 0 0 0 1 1 1 0	0 1 0 0 0 0 0 0	433 MHz

Fields:

- TXFreq3 - MSB - 0x03
- TXFreq2 - High Middle Byte - 0x02
- TXFreq1 - Low Middle Byte - 0x01
- TXFreq0 - LSB - 0x00

Recommended Value: N/A. This register should be set to the mission's transmit frequency.

Notes: While $433\,\mathrm{MHz}$ is shown as the default value of this register, the register will be programmed with the center frequency of the band to which the transmitter is configured prior to delivery.

The acceptable range for this register depends on the frequency band to which the transmitter is configured.

8.1.2 RXFreq[0-3]

Address: 0x04

Data Type: uint32_t (Hz)

Description: The RXFreq register contains the receive frequency for the Elysium as a 32-bit unsigned integer in Hz.

Diagram:

Register 8.2: RXFREQ (0x04)

RXFreq3	RXFreq2	RXFreq1	RXFreq0	
31 24	23 16	15 8	7 0	
0 0 0 1 1 0 1 1	1 1 0 0 1 1 1 1	0 0 0 0 1 1 1 0	0 1 0 0 0 0 0 0	433 MHz

Fields:

- RXFreq3 - MSB - 0x07
- RXFreq2 - High Middle Byte - 0x06
- RXFreq1 - Low Middle Byte - 0x05
- RXFreq0 - LSB - 0x04

Recommended Value: N/A. This register should be set to the mission's receive frequency.

Notes: While $433\,\mathrm{MHz}$ is shown as the default value of this register, the register will be programmed with the center frequency of the band to which the receiver is configured prior to delivery.

The acceptable range for this register depends on the frequency band to which the receiver is configured.

8.1.3 TXDev[0-3]

Address: 0x08

Data Type: uint32_t (Hz)

Description: The TXDev register contains the transmitted frequency deviation of the Elysium as a 32-bit unsigned integer in Hz.

Diagram:

Register 8.3: TXDEV (0x08)

TXDev3	TXDev2	TXDev1	TXDev0	
31 24	23 16	15 8	7 0	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 1 0 0 1 1	1 0 0 0 1 0 0 0	5 kHz

Fields:

- TXDev3 - MSB - 0x0B
- TXDev2 - High Middle Byte - 0x0A
- TXDev1 - Low Middle Byte - 0x09

- TXDev0 - LSB - 0x08

Recommended Value: Recommended values for this register are one-quarter of the bitrate (TXBR) for minimum-shift keying or one-half of the bitrate for non-minimum-shift keying.

Notes: The acceptable range for this register is $600\,\mathrm{Hz}$ to $200\,\mathrm{kHz}$.

Note that this is the deviation from carrier, i.e. $\frac{|f_{mark}-fspace|}{2}$, rather than the difference between the mark and space tones.

8.1.4 RXDev[0-3]

Address: 0x0C

Data Type: uint32_t (Hz)

Description: The RXDev register contains the expected frequency deviation of the Elysium receiver as a 32-bit unsigned integer in Hz.

Diagram:

Register 8.4: RXDev (0x0C)

RXDev3	RXDev2	RXDev1	RXDev0	
31 24	23 16	15 8	7 0	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	1 1 0 0 0 0 1 1	0 1 0 1 0 0 0 0	50 kHz

Fields:

- RXDev3 - MSB - 0x0F
- RXDev2 - High Middle Byte - 0x0E
- RXDev1 - Low Middle Byte - 0x0D
- RXDev0 - LSB - 0x0C

Recommended Value: This register should be set to the deviation of the ground station transmitter.

Notes: The acceptable range for this register is $33\,\mathrm{kHz}$ to $200\,\mathrm{kHz}$.

Note that this is the deviation from carrier, i.e. $\frac{|f_{mark}-fspace|}{2}$, rather than the difference between the mark and space tones.

8.1.5 TXSync[0-3]

Address: 0x10

Data Type: uint32_t

Description: The TXSync register contains the synchronization word transmitted at the start of each PHY-layer packet for synchronization with the ground station receiver.

Diagram:

Register 8.5: TXSync (0x10)

TXSync3	TXSync2	TXSync1	TXSync0	
31 24	23 16	15 8	7 0	
1 0 0 1 0 1 1 1	1 1 1 0 1 0 0 1	1 0 0 0 0 0 1 1	1 1 1 0 1 0 1 0	0x97E983EA

Fields:

- TXSync3 - MSB - 0x13
- TXSync2 - High Middle Byte - 0x12
- TXSync1 - Low Middle Byte - 0x11
- TXSync0 - LSB - 0x10

Recommended Value: This register should be set to a value with low autocorrelation side lobes. The value programmed by Adamant will satisfy this property. Alternatively, many Data Link Layer protocols (such as the CCSDS Space Data Link Protocols) require a particular sync value. In those cases, the Data Link Layer subsystem may override the value of this register.

Notes: Although an example value is shown here, this register will be programmed to a (semi-)random 32-bit value prior to delivery.

8.1.6 RXSync[0-4]

Address: 0x14

Data Type: uint32_t

Description: The RXSync register contains the synchronization word which the receiver detects at the start of each PHY-layer packet in order to synchronize with the ground station transmitter.

Diagram:

Register 8.6: RXSync (0x14)

RXSync3	RXSync2	RXSync1	RXSync0	
31 24	23 16	15 8	7 0	
1 0 0 0 1 0 1 0	1 1 0 1 1 0 0 0	1 0 1 1 1 0 1 1	0 0 1 0 1 0 1 0	0x8AD8BB2A

Fields:

- RXSync3 - MSB - 0x17
- RXSync2 - High Middle Byte - 0x16
- RXSync1 - Low Middle Byte - 0x15
- RXSync0 - LSB - 0x14

Recommended Value: This register should be set to a value with low autocorrelation side lobes. The value programmed by Adamant will satisfy this property. Alternatively, many Data Link Layer protocols (such as the CCSDS Space Data Link Protocols) require a particular sync value. In those cases, the Data Link Layer subsystem may override the value of this register.

Notes: Although an example value is shown here, this register will be programmed to a (semi-)random 32-bit value prior to delivery.

8.1.7 TXBR[0-3]

Address: 0x18

Data Type: uint32_t (baud)

Description: The TXBR register contains the bit rate of the transmitter as an unsigned 32-bit integer in baud, or bps. This is the *bit* rate, not necessarily the *data* rate after FEC and other overhead.

Diagram:

Register 8.7: TXBR (0x18)

TXBR3	TXBR2	TXBR1	TXBR0	
31 24	23 16	15 8	7 0	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 1 0 0 1 0	1 1 0 0 0 0 0 0	4.8 kbps

Fields:

- TXBR3 - MSB - 0x1B
- TXBR2 - High Middle Byte - 0x1A

- TXBR1 - Low Middle Byte - 0x19
- TXBR0 - LSB - 0x18

Recommended Value: N/A. This register should be set to the required transmit data rate.

Notes: The acceptable range for this register is $1.2\,\mathrm{kbps}$ to $300\,\mathrm{kbps}$.

8.1.8 RXBR[0-3]

Address: 0x1C

Data Type: uint32_t (baud)

Description: The RXBR register contains the expected bit rate of the receiver as an unsigned 32-bit integer in baud, or bps. This is the *bit* rate, not necessarily the *data* rate after FEC and other overhead.

Diagram:

Register 8.8: RXBR (0x1C)

RXBR3	RXBR2	RXBR1	RXBR0	
31 24	23 16	15 8	7 0	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 1 1 0 0 0 0 1	1 0 1 0 1 0 0 0	25 kbps

Fields:

- RXBR3 - MSB - 0x1F
- RXBR2 - High Middle Byte - 0x1E
- RXBR1 - Low Middle Byte - 0x1D
- RXBR0 - LSB - 0x1C

Recommended Value: N/A. This register should be set to the required transmit data rate.

Notes: The acceptable range for this register depends on the frequency band the receiver is configured for. For frequencies less than $680\,\mathrm{MHz}$, the acceptable range is $0.78\,\mathrm{kbps}$ to $150\,\mathrm{kbps}$. For frequencies greater than $680\,\mathrm{MHz}$, the acceptable range is $1.56\,\mathrm{kbps}$ to $200\,\mathrm{kbps}$.

8.1.9 FilterParams

Address: 0x20

Data Type: Bitfields

Description: The FilterParams register contains a number of bitfields which control the bandwidth of the receiver filter, the BT product of the transmitter's Gaussian filter (if used), and the presence or absence of an external LNA powered by the on-board bias tee.

Diagram:

Register 8.9: FILTERPARAMS (0x20)

Res.		TX_BT		RX_BW			
7	6	5	4	3			0
1	1	0	0	0	0	1	1

Fields:

- Res. - Reserved. These bits are ignored. - 0x20.6
- TX_BT - 00: Disabled. 01: 1.0. 10: 0.5. 11: 0.3. - 0x20.4
- RX_BW - $BW = 25\,\mathrm{khz} * (RX_B W + 1)$ - 0x20.0

Recommended Value: Gaussian filtering should be used if necessary to meet bandwidth requirements. Receiver bandwidth should be set as close as possible to the 99% energy bandwidth of the expected signal plus any uncompensated Doppler shift. The 99% energy bandwidth of an FSK signal is approximately $2 * [F_{dev} + \frac{BR}{2}]$.

Notes: The reserved bits are ignored and may safely be set to any value.

The TX_BT field is an enumeration of BT settings for the transmitter's Gaussian filter. A value of 00 disables the Gaussian filter. A value of 01 sets its BT to 1.0. A value of 10 sets BT to 0.5, while a value of 11 sets it to 0.3. For more information, please consult a reference on Gaussian filtering.

The RX_BW field sets the receive filter's single-sided bandwidth according to the formula $BW = 25\,\mathrm{kHz} * (RX_B W + 1)$, such that 0000 represents $25\,\mathrm{kHz}$ and 1111 represents $400\,\mathrm{kHz}$. The default value of 0011 represents $100\,\mathrm{kHz}$.

8.1.10 OutputPower

Address: 0x21

Data Type: uint8_t

Description: The OutputPower register programs the output power of the transmitter according to the formula shown below.

Diagram:

Register 8.10: OutputPower (0x21)

OutputPower	
7 0	
0 0 0 0 1 0 1 0	30 dBm

Fields:

- OutputPower - $P_{OUT} = [10 + (0.2 * OutputPower)]dBm$ - 0x21

Recommended Value: N/A. This should be set to the lowest value which provides adequate link margin under the conditions of the mission.

Notes: The OutputPower register sets the transmitter's output power according to the formula $P_{OUT} = [10 + (0.2 * OutputPower)]dBm$. This is the RF output power after the gain of the amplifier and all other effects.

The acceptable range for this register is 0 $(10\,\mathrm{dBm})$ to 135 $(37\,\mathrm{dBm})$.

Not all physical configurations of the Elysium can produce all output powers within this acceptable range. When the register is set to a value outside of the achievable range but within the acceptable range, the built in closed loop control of the output power will cause the actual output power to be clipped to the nearest achievable value.

8.1.11 UARTBaud[0-3]

Address: 0x22

Data Type: uint32_t (baud)

Description: The UARTBaud register contains the programmed baud rate of the onboard UART as an unsigned 32-bit integer in baud (or bps).

Diagram:

Register 8.11: UARTBaud (0x22)

UARTBaud3	UARTBaud2	UARTBaud1	UARTBaud0	
31 24	23 16	15 8	7 0	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 1	1 1 0 0 0 0 1 0	0 0 0 0 0 0 0 0	115200 baud

Fields:

- UARTBaud3 - MSB - 0x25
- UARTBaud2 - High Middle Byte - 0x24
- UARTBaud1 - Low Middle Byte - 0x23
- UARTBaud0 - LSB - 0x22

Recommended Value: Common baud rates include 9600 baud, 115.2 kbaud, and 1 Mbaud.

Notes: The acceptable range for this register is $1\,\mathrm{baud}$ to $16\,\mathrm{Mbaud}$. Achievable rates depend on the configuration of the spacecraft (length of cabling runs, strength of drivers, termination, etc).

See also the Autobaud field of the UARTParams register.

8.1.12 UARTParams

Address: 0x26

Data Type: Bitfields

Description: The UARTParams register contains a number of bitfields which control the parity bit (if any) used by the UART, the number of stop bits, and whether or not the UART makes use of Autobaud functionality.

Diagram:

Register 8.12: UARTParams (0x26)

Res.	Autobaud	TwoStop	Parity
7 4	3	2	1 0
1 1 1 1	0	0	0 0

Fields:

- Res. - Reserved. These bits are ignored. - 0x26.4
- Autobaud - Setting this bit enables LIN-style Autobaud functionality - 0x26.3
- TwoStop - When this bit is set, 2 stop bits are used, otherwise 1 is used - 0x26.2
- Parity - 00: No parity bit. 01: Even parity. 10: Odd parity. 11: Illegal value. - 0x26.0

Recommended Value: Autobaud presents certain risks and should be used with caution. The most common UART format is 8N1 (no double-stop, no parity). Parity may be required in certain uncommon cases.

Notes: The reserved bits are ignored and may safely be set to any value.

When the Autobaud bit is set, LIN-style Autobaud functionality is enabled on the UART. In this mode, the UART is initially programmed with the value of the UARTBaud register as its data rate. It then waits for a synchronization sequence consisting of a break (11 to 21 continuous zeros/spaces) and a synch field (0x55). These fields are used to determine the baud rate used by the transmitter. The range of data rates which can be detected in this mode is $244\,\mathrm{baud}$ to $1\,\mathrm{Mbaud}$. The Elysium cannot send UART data while receiving the break/synch fields.

When the TwoStop bit is set, two stop bits are used by the UART for byte framing (e.g. format 8N2). When the TwoStop bit is not set, only one stop is used (e.g. 8N1).

The Parity bitfield contains an enumeration of the parity bit, if any, used by the UART for byte framing. The value 00 represents no parity bit (e.g. 8N1), the value 01 represents an even parity bit (e.g. 8E1), and the value 10 represents an odd parity bit (e.g. 8O1). The value 11 is illegal and outside the acceptable range for this register - it will be changed to 00 if requested.

8.1.13 FaultResponse

Address: 0x27

Data Type: Bitfields

Description: The FaultResponse register contains a number of bitfields which control the activation of certain fault response handlers. See Section 10 for more details.

Diagram:

Register 8.13: FaultResponse (0x27)

Res.	PATempFault	HSTempFault	BoardTempFault	GroundFaultBeacon	GroundFaultFallback	SCCommFaultRst	SCCommFaultGPO	
6	6	5	4	3	2	1	0	
1	1	1	1	1	1	1	1	All Responses Enabled

Fields:

- Res. - Reserved. This bit is ignored. - 0x27.7
- PATempFault - Enables power amplifier overtemp fault responses - 0x27.6
- HSTempFault - Enables heatsink overtemp fault responses - 0x27.5
- BoardTempFault - Enables MCU overtemp fault responses - 0x27.4
- GroundFaultBeacon - Enables ground contact lost beacon fault responses - 0x27.3
- GroundFaultFallback - Enables ground contact lost fallback fault responses - 0x27.2
- SCCommFaultRst - Enables spacecraft communication lost fault responses (internal) - 0x27.1

- SCCommFaultGPO - Enables spacecraft communication lost fault responses (external) - 0x27.0

Recommended Value: The overtemperature fault responses should always be enabled. It is usually also a good idea to dedicate a Register Bank to the maximum-link-margin configuration supported by the mission for use with the GroundFaultFallback fault response.

Notes: The reserved bit is ignored and may safely be set to any value.

When the PATempFault bit is set, fault handling of power amplifier overtemperature events is enabled. The Elysium will automatically power off the transmitter when the PA temperature reaches the value set by the PATempERR register. The transmitter will remain off for a duration set by the OTFaultTime register. This allows a chance for the power amplifier to dissipate heat and recover from the overtemperature condition.

When the HSTempFault bit is set, fault handling of heatsink overtemperature events is enabled. The Elysium will automatically power off the transmitter when the heatsink temperature reaches the value set by the HSTempERR register. The transmitter will remain off for a duration set by the OTFaultTime register. This allows a chance for the heatsink to dissipate heat and recover from the overtemperature condition.

When the BoardTempFault bit is set, fault handling of MCU overtemperature events is enabled. The Elysium will automatically power off the transmitter when the MCU temperature reaches the value set by the BoardTempErr register. The transmitter will remain off for a duration set by the OTFaultTime register. This allows a chance for the radio to dissipate heat and recover from the overtemperature condition.

When the GroundFaultBeacon bit is set, beacon mode fault handling of ground contact loss events is enabled. When the Elysium has not received any packets from the ground (either passthrough packets or Elyisum command packets) for the interval specified by the GFTime register, the radio will begin broadcasting beacon messages at the interval specified in the BeaconTime register. If the GroundFaultFallback bit is also set, the fallback configuration is loaded before the beacon messages begin. The exact format of beacon messages depends on the Data Link Layer - for example, the SDLP Data Link Layer broadcasts Only Idle Data (OID) frames as beacon frames.

When the GroundFaultFallback bit is set, fallback fault handling of ground contact loss events is enabled. When the Elysium has not received any packets from the ground (either passthrough packets or Elyisum command packets) for the interval specified by the GFTime register, the configuration stored in the register bank specified by the GFBank register will be loaded into the working memory of the Elysium. This allows for a failsafe configuration to be maintained in the event of a botched configuration change.

When the SCCommFaultRst bit is set, internal fault handling of spacecraft communication loss events is enabled. When the Elysium has not received any packets from the spacecraft (either passthrough packets or Elyisum command packets) for the interval specified by the SCCommTime register, the UART baud rate specified by the SCCommRate register will be loaded into the working memory of the Elysium. This allows for a failsafe configuration to be maintained in the event of a botched configuration change. The Elysium will also be reset in an attempt to clear any error condition of the UART hardware.

When the SCCommFaultGPO bit is set, external fault handling of spacecraft communication loss events is enabled. When the Elysium has not received any packets from the spacecraft (either passthrough packets or Elyisum command packets) for the interval specified by the SCCommTime register, the UART baud rate specified by the SCCommRate register will be loaded into the working memory of the Elysium. This allows for a failsafe configuration to be maintained in the event of a botched configuration change. The Elysium will also toggle the GPO pin for $1\,\mathrm{ms}$ in an attempt to reset external hardware which may be in a hung state.

The fault responses for overcurrent and undervoltage events are non-optional. These conditions present serious risks to the hardware and almost always mean that something very bad is in the process of happening. If you really believe that you need to disable them, please contact Adamant for more information.

See Section 10 for more information on fault responses.

8.1.14 ErrorReport[0-1]

Address: 0x28

Data Type: Bitfields

Description: The ErrorReport register contains a number of bitfields which control the reporting of certain errors.

Diagram:

Register 8.14: ErrorReport (0x28)

PATempRpt	HSTempRpt	BoardTempRpt	GFRpt	SCCommRpt	LNAOCRpt	PAOCRpt	UVRpt	RegRpt	OpRpt	LengthRpt	FCSRpt	NoReplyRpt	ChanOverRpt	ResetRpt	UARTRpt	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	All Reports Enabled

Fields:

- PATempRpt - Enables reporting of PA overtemperature errors - 0x29.7
- HSTempRpt - Enables reporting of heat sink overtemperature errors - 0x29.6
- BoardTempRpt - Enables reporting of MCU overtemperature errors - 0x29.5
- GFRpt - Enables reporting of ground comm loss errors - 0x29.4
- SCCommRpt - Enables reporting of spacecraft comm loss errors - 0x29.3
- LNAOCRpt - Enables reporting of LNA overcurrent errors - 0x29.2
- PAOCRpt - Enables reporting of PA overcurrent errors - 0x29.1
- UVRpt - Enables reporting of digital undervoltage errors - 0x29.0
- RegRpt - Enables reporting of register value clipping for Elysium registers - 0x28.7
- OpRpt - Enables reporting of invalid opcode errors for Elysium commands - 0x28.6
- LengthRpt - Enables reporting of packet length errors for Elysium commands - 0x28.5
- FCSRpt - Enables reporting of FCS errors for Elysium commands - 0x28.4

- NoReplyRpt - Enables reporting of command value errors for commands without the Reply Requested bit set - 0x28.3
- SubOverRpt - Enables reporting of channel or event subscription data being overwritten - 0x28.2
- ResetRpt - Enables reporting of Elysium resets - 0x28.1
- UARTRpt - Enables reporting of UART framing errors - 0x28.0

Recommended Value: The overcurrent, undervoltage, and PA overtemperature errors represent the largest risk to hardware and should have first priority when deciding what to report to the flight computer.

Notes: The reserved bits are ignored and may safely be set to any value.

When the PATempRpt bit is set, a measured temperature at the power amplifier thermistor which exceeds the value set in the PATempWARN register will produce an error of WARNING priority, while a measured temperature exceeding the value set in the PATempERR register will produce an error of ERROR priority.

When the HSTempRpt bit is set, a measured temperature at the heatsink thermistor which exceeds the value set in the HSTempWARN register will produce an error of WARNING priority, while a measured temperature exceeding the value set in the HSTempERR register will produce an error of ERROR priority.

When the BoardTempRpt bit is set, a measured temperature at the MCU which exceeds the value set in the BoardTempWARN register will produce an error of WARNING priority, while a measured temperature exceeding the value set in the BoardTempERR register will produce an error of ERROR priority.

When the GFRpt bit is set, a loss of communication with the ground station lasting longer than the duration set in the GFTime register will produce an error with the priority level defined in the GFLvl register.

When the SCCommRpt bit is set, a loss of communication with the rest of the spacecraft lasting longer than the duration set in the SCCommTime register will produce an error with the priority level defined in the SCCommLvl register.

When the LNAOCRpt bit is set, LNA overcurrent errors will produce an error with priority level CRITICAL *after* the appropriate fault response has been taken. See Section 10 for more details.

When the PAOCRpt bit is set, PA overcurrent errors will produce an error with priority level CRITICAL *after* the appropriate fault response has been taken. See Section 10 for more details.

When the UVRpt bit is set, microcontroller undervoltage errors (usually caused by a latch-up condition) will produce an error with priority level CRITICAL. See Section 10 for more details.

When the RegRpt bit is set, any invalid value which the user attempts to assign to an Elysium register using the SetRegs command will, in addition to being clipped to the nearest acceptable value, produce an error with the priority level defined in the RegErrLvl register.

When the OpRpt bit is set, any invalid opcode present in an Elysium command packet's header will, in addition to discarding the command packet, produce an error with the priority level defined in the OpErrLvl register.

When the LengthRpt bit is set, any mismatch between the Length field of an Elysium command packet's header and the expected length of the command will, in addition to discarding the command packet, produce an error with the priority level defined in the LengthErrLvl register.

When the FCSRpt bit is set, any mismatch between the FCS field of an Elysium command packet and the calculated CRC of the packet will, in addition to discarding the command packet, produce an error with the priority level defined in the FCSLvl register.

When the NoReplyRpt bit is set, any command value error which would produce a FAILURE message if the Reply Requested bit were set, but which does not because the Reply Requested bit is not set on the current packet, will instead produce an error with the priority level defined in the NRErrLvl register.

When the SubOverRpt bit is set, anytime an existing Channel subscription is overwritten by a new ChannelSub command without first being deleted by a ChannelUnsub command, or an Event subscription is overwritten by a new EventSub command without first being deleted by a EventUnsub command, the Elysium will report an error with the priority level defined in the SubOverLvl register.

When the ResetRpt bit is set, any spurious resets (resets not associated with Reset or InstallFW commands) will produce an error report with the priority level defined in the ResetErrLvl register.

When the UARTRpt bit is set, any UART errors (framing, overrun, parity) will produce an error with the priority level defined in the UARTErrLvl register.

8.1.15 ResetErrLvl

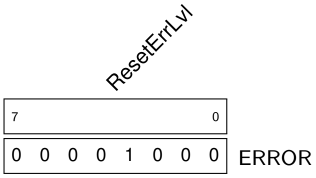
Address: 0x2A

Data Type: Priority Enumeration

Description: The ResetErrLvl register controls the priority level of spurious reset errors, if enabled by the ResetRpt bit of the ErrorReport register.

Diagram:

Register 8.15: ResetErrLvl (0x2A)



Fields:

- ResetErrLvl - Priority level of spurious reset error reports - 0x2A

Recommended Value: ERROR

Notes: The acceptable values for this register are the valid values of the Priority Enumeration data type, a one-hot encoding using bits 0 through 4.

8.1.16 UARTErrLvl

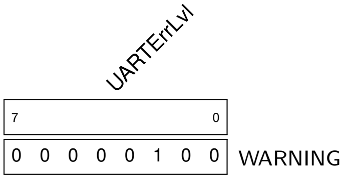
Address: 0x2B

Data Type: Priority Enumeration

Description: The UARTErrLvl register controls the priority level of UART error reports, if enabled by the UARTRpt bit of the ErrorReport register.

Diagram:

Register 8.16: UARTErrLvl (0x2B)



Fields:

- UARTErrLvl - Priority level of UART error reports - 0x2B

Recommended Value: WARNING

Notes: The acceptable values for this register are the valid values of the Priority Enumeration data type, a one-hot encoding using bits 0 through 4.

8.1.17 SubOverLvl

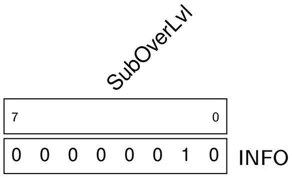
Address: 0x2C

Data Type: Priority Enumeration

Description: The SubOverLvl register controls the priority level of channel or event subscription overwriting errors, if enabled by the SubOverRpt bit of the ErrorReport register.

Diagram:

Register 8.17: SubOverLvl (0x2C)



Fields:

- SubOverLvl - Priority level of channel subscription overwriting reports - 0x2C

Recommended Value: INFO

Notes: The acceptable values for this register are the valid values of the Priority Enumeration data type, a one-hot encoding using bits 0 through 4.

8.1.18 NRErrLvl

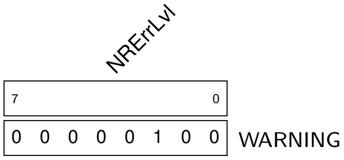
Address: 0x2D

Data Type: Priority Enumeration

Description: The NRErrLvl register controls the priority level of command data value errors, if enabled by the NoReplyRpt bit of the ErrorReport register.

Diagram:

Register 8.18: NRErrLvl (0x2D)



Fields:

- NRErrLvl - Priority level of command data value error reports - 0x2D

Recommended Value: WARNING

Notes: The acceptable values for this register are the valid values of the Priority Enumeration data type, a one-hot encoding using bits 0 through 4.

8.1.19 FCSLvl

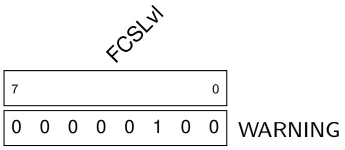
Address: 0x2E

Data Type: Priority Enumeration

Description: The FCSLvl register controls the priority level of command packet FCS errors, if enabled by the FCSRpt bit of the ErrorReport register.

Diagram:

Register 8.19: FCSLvl (0x2E)



Fields:

- FCSLvl - Priority level of command packet FCS error reports - 0x2E

Recommended Value: WARNING

Notes: The acceptable values for this register are the valid values of the Priority Enumeration data type, a one-hot encoding using bits 0 through 4.

8.1.20 LengthErrLvl

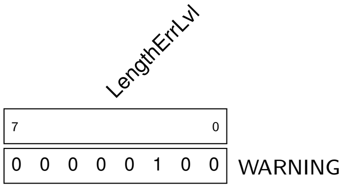
Address: 0x2F

Data Type: Priority Enumeration

Description: The LengthErrLvl register controls the priority level of command packet length errors, if enabled by the LengthRpt bit of the ErrorReport register.

Diagram:

Register 8.20: LengthErrLvl (0x2F)



Fields:

- LengthErrLvl - Priority level of command packet length error reports - 0x2F

Recommended Value: WARNING

Notes: The acceptable values for this register are the valid values of the Priority Enumeration data type, a one-hot encoding using bits 0 through 4.

8.1.21 OpErrLvl

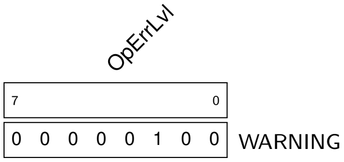
Address: 0x30

Data Type: Priority Enumeration

Description: The OpErrLvl register controls the priority level of command packet length errors, if enabled by the OpRpt bit of the ErrorReport register.

Diagram:

Register 8.21: OpErrLvl (0x30)



Fields:

- OpErrLvl - Priority level of invalid opcode error reports - 0x30

Recommended Value: WARNING

Notes: The acceptable values for this register are the valid values of the Priority Enumeration data type, a one-hot encoding using bits 0 through 4.

8.1.22 RegErrLvl

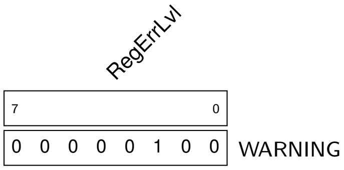
Address: 0x31

Data Type: Priority Enumeration

Description: The RegErrLvl register controls the priority level of register value clipping errors, if enabled by the RegRpt bit of the ErrorReport register.

Diagram:

Register 8.22: RegErrLvl (0x31)



Fields:

- RegErrLvl - Priority level of register clipping error reports - 0x31

Recommended Value: WARNING

Notes: The acceptable values for this register are the valid values of the Priority Enumeration data type, a one-hot encoding using bits 0 through 4.

8.1.23 SCCommLvl

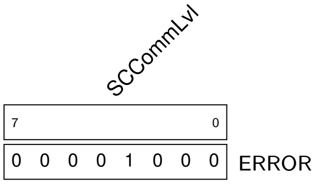
Address: 0x32

Data Type: Priority Enumeration

Description: The SCCommLvl register controls the priority level of spacecraft communication loss errors, if enabled by the SCCommRpt bit of the ErrorReport register.

Diagram:

Register 8.23: SCCommLvl (0x32)



Fields:

- SCCommLvl - Priority level of spacecraft communication loss error reports - 0x32

Recommended Value: ERROR

Notes: The acceptable values for this register are the valid values of the Priority Enumeration data type, a one-hot encoding using bits 0 through 4.

Fault reporting and fault handling are independent - a Spacecraft Communication Loss Fault may be reported without activating the fault handler, or vice versa.

8.1.24 GFLvl

Address: 0x33

Data Type: Priority Enumeration

Description: The GFLvl register controls the priority level of ground communication loss errors, if enabled by the GFRpt bit of the ErrorReport register.

Diagram:

Register 8.24: GFLvl (0x33)

GFLvl	
7 0	
0 0 0 0 1 0 0 0	ERROR

Fields:

- GFLvl - Priority level of ground communication loss error reports - 0x33

Recommended Value: ERROR

Notes: The acceptable values for this register are the valid values of the Priority Enumeration data type, a one-hot encoding using bits 0 through 4.

Fault reporting and fault handling are independent - a Ground Fault may be reported without activating the fault handler, or vice versa.

8.1.25 SCCommTime[0-4]

Address: 0x34

Data Type: uint32_t (seconds)

Description: The SCCommTime register contains the length of time the Elysium will wait without hearing from the rest of the spacecraft before activating the Spacecraft Communication Lost fault response, if enabled by the SCCommFaultRst or SCCommFaultGPO bits of the FaultResponse register.

Diagram:

Register 8.25: SCCommTime (0x34)

SCCommTime3	SCCommTime2	SCCommTime1	SCCommTime0	
31 24	23 16	15 8	7 0	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 1 0	1 0 1 0 0 0 1 1	0 0 0 0 0 0 0 0	48 Hours

Fields:

- SCCommTime3 - MSB - 0x37
- SCCommTime2 - High Middle Byte - 0x36
- SCCommTime1 - Low Middle Byte - 0x35
- SCCommTime0 - LSB - 0x34

Recommended Value: N/A. Depends on mission configuration.

Notes: See Section 10 for more details of the Spacecraft Communication Lost fault handler.

8.1.26 SCCommBaud[0-3]

Address: 0x38

Data Type: uint32_t (baud)

Description: The SCCommBaud register contains the programmed baud rate of the onboard UART which will be configured by the Spacecraft Communication Lost fault reponse, if enabled by the SCCommRst or SCcommFAultGPO bits of the FaultResponse register. The baud rate is stored as an unsigned 32-bit integer in baud (or bps).

Diagram:

Register 8.26: SCCommBaud (0x38)

SCCommBaud3	SCCommBaud2	SCCommBaud1	SCCommBaud0	
31 24	23 16	15 8	7 0	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 1 0 0 1 0 1	1 0 0 0 0 0 0 0	9600 baud

Fields:

- SCCommBaud3 - MSB - 0x3B
- SCCommBaud2 - High Middle Byte - 0x3A
- SCCommBaud1 - Low Middle Byte - 0x39
- SCCommBaud0 - LSB - 0x38

Recommended Value: 9600 baud is a very common fallback baud rate.

Notes: The acceptable range for this register is $1\,\mathrm{baud}$ to $16\,\mathrm{Mbaud}$. Achievable rates depend on the configuration of the spacecraft (length of cabling runs, strength of drivers, termination, etc).

See also the Autobaud field of the UARTParams register.

8.1.27 GFTime[0-4]

Address: 0x3C

Data Type: uint32_t (seconds)

Description: The GFTime register contains the length of time the Elysium will wait without hearing from the ground station before activating the Ground Fault fault response, if enabled by the GroundFaultBeacon or GroundFaultFallback bit of the FaultResponse register.

Diagram:

Register 8.27: GFTime (0x3C)

GFTime3	GFTime2	GFTime1	GFTime0	
31 24	23 16	15 8	7 0	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 1 0	1 0 1 0 0 0 1 1	0 0 0 0 0 0 0 0	48 Hours

Fields:

- GFTime3 - MSB - 0x3F
- GFTime2 - High Middle Byte - 0x3E
- GFTime1 - Low Middle Byte - 0x3D
- GFTime0 - LSB - 0x3C

Recommended Value: N/A. Depends on mission orbit and ground station locations.

Notes: See Section 10 for more details of the Ground Fault fault handler.

8.1.28 BeaconTime[0-4]

Address: 0x40

Data Type: uint32_t (seconds)

Description: The BeaconTime register contains the interval at which the Elysium will transmit beacon frames when in Beacon Mode due to the Ground Fault fault response, if enabled by the GroundFaultBeacon bit of the FaultResponse register.

Diagram:

Register 8.28: BeaconTime (0x40)

BeaconTime3	BeaconTime2	BeaconTime1	BeaconTime0	
31 24	23 16	15 8	7 0	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 1	0 0 1 0 1 1 0 0	5 Minutes

Fields:

- BeaconTime3 - MSB - 0x43
- BeaconTime2 - High Middle Byte - 0x42
- BeaconTime1 - Low Middle Byte - 0x41
- BeaconTime0 - LSB - 0x40

Recommended Value: N/A. Depends on mission orbit and ground station locations.

Notes: See Section 10 for more details of the Ground Fault fault handler.

8.1.29 GFBank

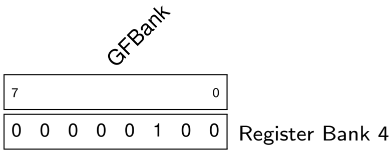
Address: 0x44

Data Type: uint8_t

Description: The GFBank register contains the index of the Register Bank whose configuration will be loaded by the Ground Fault fault response, if enabled by the GroundFaultFallback bit of the FaultResponse register.

Diagram:

Register 8.29: GFBank (0x44)



Fields:

- GFBank - Fallback Register Bank for Ground Fault fault response- 0x44

Recommended Value: Often Register Bank 4 is used as a “safe” configuration bank.

Notes: See Section 10 for more details of the Ground Fault Response fault handler.

8.1.30 OTFaultTime

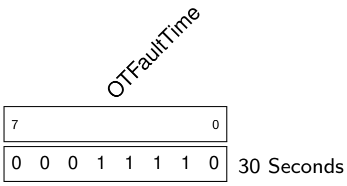
Address: 0x45

Data Type: uint8_t (seconds)

Description: The OTFaultTime register contains the length of time in seconds for which power will be removed from the transmitter and PA when the Overtemp Fault Response is activated, if enabled by the PATempFault, HSTempFault, or BoardTempFault bits of the FaultResponse register.

Diagram:

Register 8.30: OTFAULTTIME (0x45)



Fields:

- OTFaultTime - Duration of overtemp fault response - 0x45

Recommended Value: Depends on thermal design - 30 seconds is usually sufficient.

Notes: See Section 10 for more details of the Overtemp Fault Response fault handler.

8.1.31 BoardTempWARN

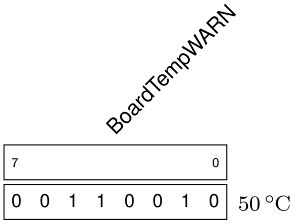
Address: 0x46

Data Type: int8_t (°C)

Description: The BoardTempWARN register contains the temperature which, when exceeded by the MCU, will cause a WARNING error report to be generated if enabled by the BoardTempRpt bit in the ErrorReport register.

Diagram:

Register 8.31: BOARDTEMPWARN (0x46)



Fields:

- BoardTempWARN - MCU WARNING threshold temperature - 0x46

Recommended Value: Depends on thermal design - 50 °C is usually reasonable.

Notes: The acceptable range for this register is from -80 °C to 125 °C. The recommended operating range for the Elysium is from -40 °C to 85 °C.

8.1.32 BoardTempERR

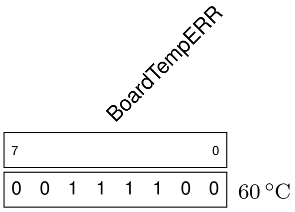
Address: 0x47

Data Type: int8_t (°C)

Description: The BoardTempERR register contains the temperature which, when exceeded by the MCU, will cause an ERROR error report to be generated if enabled by the BoardTempRpt bit in the ErrorReport register. Exceeding this temperature will also activate the MCU Overtemperature Fault handler if enabled by the BoardTempFault bit in the FaultResponse register.

Diagram:

Register 8.32: BoardTempERR (0x47)



Fields:

- BoardTempERR - MCU ERROR threshold temperature - 0x47

Recommended Value: Depends on thermal design - 60 °C is usually reasonable.

Notes: The acceptable range for this register is from -80 °C to 125 °C. The recommended operating range for the Elysium is from -40 °C to 85 °C.

8.1.33 HSTempWARN

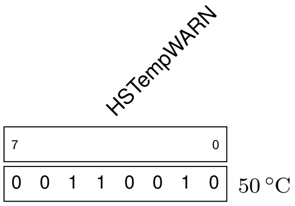
Address: 0x48

Data Type: int8_t (°C)

Description: The HSTempWARN register contains the temperature which, when exceeded by the heatsink, will cause a WARNING error report to be generated if enabled by the HSTempRpt bit in the ErrorReport register.

Diagram:

Register 8.33: HSTempWARN (0x48)



Fields:

- HSTempWARN - Heatsink WARNING threshold temperature - 0x48

Recommended Value: Depends on thermal design - 50 °C is usually reasonable.

Notes: The acceptable range for this register is from -80 °C to 125 °C. The recommended operating range for the Elysium is from -40 °C to 85 °C.

8.1.34 HSTempERR

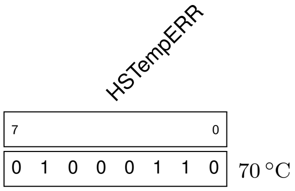
Address: 0x49

Data Type: int8_t (°C)

Description: The HSTempERR register contains the temperature which, when exceeded by the heatsink, will cause an ERROR error report to be generated if enabled by the HSTempRpt bit in the ErrorReport register. Exceeding this temperature will also activate the Heatsink Overtemperature Fault handler if enabled by the HSTempFault bit in the FaultResponse register.

Diagram:

Register 8.34: HSTempERR (0x49)



Fields:

- HSTempERR - Heatsink ERROR threshold temperature - 0x49

Recommended Value: Depends on thermal design - 70 °C is usually reasonable.

Notes: The acceptable range for this register is from -80 °C to 125 °C. The recommended operating range for the Elysium is from -40 °C to 85 °C.

8.1.35 PATempWARN

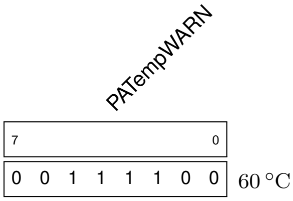
Address: 0x4A

Data Type: int8_t ($^{\circ}\mathrm{C}$)

Description: The PATempWARN register contains the temperature which, when exceeded by the power amplifier, will cause a WARNING error report to be generated if enabled by the PATempRpt bit in the ErrorReport register.

Diagram:

Register 8.35: PATEMPWARN (0x4A)



Fields:

- PATempWARN - PA WARNING threshold temperature - 0x4A

Recommended Value: Depends on thermal design - $60\,^{\circ}\mathrm{C}$ is usually reasonable.

Notes: The acceptable range for this register is from $-80\,^{\circ}\mathrm{C}$ to $125\,^{\circ}\mathrm{C}$. The recommended operating range for the Elysium is from $-40\,^{\circ}\mathrm{C}$ to $85\,^{\circ}\mathrm{C}$.

8.1.36 PATempERR

Address: 0x4B

Data Type: int8_t ($^{\circ}\mathrm{C}$)

Description: The PATempERR register contains the temperature which, when exceeded by the power amplifier, will cause an ERROR error report to be generated if enabled by the PATempRpt bit in the ErrorReport register. Exceeding this temperature will also activate the Heatsink Overtemperature Fault handler if enabled by the PATempFault bit in the FaultResponse register.

Diagram:

Register 8.36: PATempERR (0x4B)

PATempERR	
7 0	
0 1 0 0 1 0 1 1	75 °C

Fields:

- PATempERR - PA ERROR threshold temperature - 0x4B

Recommended Value: Depends on thermal design - $75\,°\mathrm{C}$ is usually reasonable.

Notes: The acceptable range for this register is from $-80\,°\mathrm{C}$ to $125\,°\mathrm{C}$. The recommended operating range for the Elysium is from $-40\,°\mathrm{C}$ to $85\,°\mathrm{C}$.

8.1.37 ErrRptLvl

Address: 0x4C

Data Type: Priority Enumeration Mask

Description: The ErrRptLvl register contains the mask of priority levels which will be reported out as error events on the UART. For example, if the INFO and ERROR bits are set, any INFO-priority or ERROR-priority errors will be reported as errors on the UART, however, any WARNING-priority or CRITICAL-priority errors will not be transmitted.

Diagram:

Register 8.37: ErrRptLvl (0x4C)

Res.	CRITICAL	ERROR	WARNING	INFO	DEBUG	
7 5	4	3	2	1	0	
1 1 1	1	1	1	0	0	CRITICAL |ERROR |WARNING

Fields:

- Res. - Reserved. These bits are ignored. - 0x4C.5
- CRITICAL - CRITICAL priority mask bit - 0x4C.4
- ERROR - ERROR priority mask bit - 0x4C.3
- WARNING - WARNING priority mask bit - 0x4C.2

- INFO - INFO priority mask bit - 0x4C.1
- DEBUG - DEBUG priority mask bit - 0x4C.0

Recommended Value: Depends on the ability of the spacecraft to correct errors. WARNING and above is usually appropriate.

Notes: The reserved bits are ignored and may safely be set to any value.

Each of the bits named for a priority, e.g. CRITICAL or ERROR, may be set to allow errors of that priorty to be reported. Errors with a priority whose corresponding bit is not set will not be reported.

See Section 10 for more information on how such error messages are reported.

8.1.38 ErrLogLvl

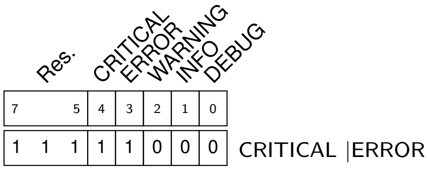
Address: 0x4D

Data Type: Priority Enumeration Mask

Description: The ErrLogLvl register contains the mask of priority levels which will be logged to the Elysium's internal telemetry memory. For example, if the INFO and ERROR bits are set, any INFO-priority or ERROR-priority errors will be logged to on-board memory, however, any WARNING-priority or CRITICAL-priority errors will not be logged. Logged errors can be extracted using the standard GetTelem command.

Diagram:

Register 8.38: ErrLogLvl (0x4D)



Res.	CRITICAL	ERROR	WARNING	INFO	DEBUG	
7 5	4	3	2	1	0	
1 1 1	1	1	0	0	0	CRITICAL |ERROR

Fields:

- Res. - Reserved. These bits are ignored. - 0x4D.5
- CRITICAL - CRITICAL priority mask bit - 0x4D.4
- ERROR - ERROR priority mask bit - 0x4D.3
- WARNING - WARNING priority mask bit - 0x4D.2
- INFO - INFO priority mask bit - 0x4D.1
- DEBUG - DEBUG priority mask bit - 0x4D.0

Recommended Value: Depends on the frequency of downlink or the ability of the spacecraft to analyze telemetry. ERROR and above is usually appropriate.

Notes: The reserved bits are ignored and may safely be set to any value.

Each of the bits named for a priority, e.g. CRITICAL or ERROR, may be set to allow errors of that priorty to be logged. Errors with a priority whose corresponding bit is not set will not be logged.

See Section 10 for more information on the format of logged error telemetry.

8.1.39 ErrRptAddr[0-1]

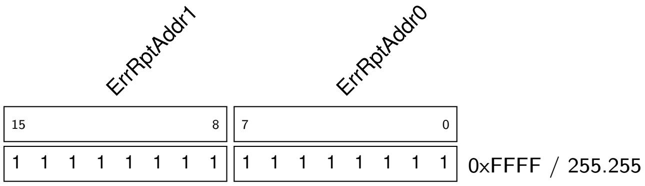
Address: 0x4E

Data Type: uint16_t

Description: The ErrRptAddr register contains the address (in the same format as the Reply Address used by commands) to which errors which pass the ErrRptLvl mask will be reported.

Diagram:

Register 8.39: ErrRptAddr (0x4E)



Fields:

- ErrRptAddr1 - MSB - 0x4F
- ErrRptAddr0 - LSB - 0x4E

Recommended Value: The address of the flight computer, if applicable, otherwise the address of the ground station.

Notes: The value of ErrRptAddr is interpreted by the Network Layer in the same way as the Reply Address of Elysium commands.

8.1.40 SrcAddr[0-1]

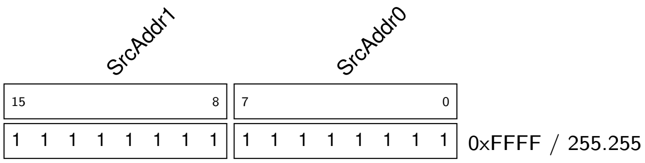
Address: 0x50

Data Type: uint16_t

Description: The SrcAddr register contains the address (in the same format as the Reply Address used by commands) of the Elysium itself. It is included in Reply packets to allow differentiation between multiple Elysium radios in a single system.

Diagram:

Register 8.40: SRCADDR (0x50)



Fields:

- SrcAddr1 - MSB - 0x51
- SrcAddr0 - LSB - 0x50

Recommended Value: N/A. Depends on mission requirements and Network Layer.

Notes: The value of SrcAddr is interpreted by the Network Layer in the same way as the Reply Address of Elysium commands.

8.1.41 ChanDefaultAddr[0-1]

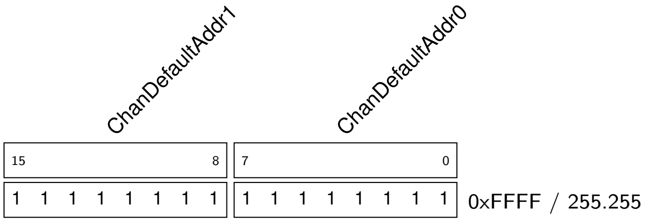
Address: 0x52

Data Type: uint16_t

Description: The ChanDefaultAddr register contains the address (in the same format as the Reply Address used by commands) to which Channel data should be sent by default.

Diagram:

Register 8.41: CHANDEFAULTADDR (0x52)



Fields:

- ChanDefaultAddr1 - MSB - 0x53
- ChanDefaultAddr0 - LSB - 0x52

Recommended Value: The address of the flight computer, if applicable.

Notes: The value of ChanDefaultAddr is interpreted by the Network Layer in the same way as the Reply Address of Elysium commands.

8.1.42 EventDefaultAddr[0-1]

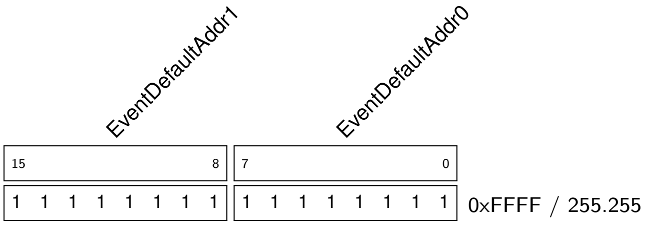
Address: 0x54

Data Type: uint16_t

Description: The EventDefaultAddr register contains the address (in the same format as the Reply Address used by commands) to which Event reports should be sent by default.

Diagram:

Register 8.42: EventDefaultAddr (0x54)



Fields:

- EventDefaultAddr1 - MSB - 0x55
- EventDefaultAddr0 - LSB - 0x54

Recommended Value: The address of the flight computer, if applicable.

Notes: The value of EventDefaultAddr is interpreted by the Network Layer in the same way as the Reply Address of Elysium commands.

8.2 Special Registers

This section defines the special registers listed in Table 11, which exist only in Register Bank 0 and cannot be directly modified.

The address range from 0x60 to 0x7F is reserved for the Special Registers.

Table 11: Special Registers

Address	*Name*	*Description*
0x60	BootCount	Elysium Boot Count
0x61	GPOState	GPO Pin State
0x62	CRITErrors	Number of CRITICAL Errors
0x63	ERRErrors	Number of ERROR Errors
0x64	WARNErrors	Number of WARNING Warnings

Table 11: Special Registers

Address	*Name*	*Description*
0x65	INFOErrors	Number of INFO Infos
0x66	DEBUGErrors	Number of DEBUG Debugs
0x67	LastError	Most Recent Error ID
0x68	LastEvent	Most Recent Event ID
0x69	ActiveBank	Most Recently Loaded Register Bank
0x6A	TelemIndex	Current Telemetry Slot Index
0x6B	Padding	Alignment Padding Register
0x6C	Uptime0	Time Since Last Reset LSB
0x6D	Uptime1	Time Since Last Reset Low Middle Byte
0x6E	Uptime2	Time Since Last Reset High Middle Byte
0x6F	Uptime3	Time Since Last Reset MSB
0x70	RFConfig0	RF Configuration Bitfields LSB
0x71	RFConfig1	RF Configuration Bitfields MSB
0x72	NetworkLayer	Network Layer ID
0x73	DataLinkLayer	Data Link Layer ID
0x74	MissionTime0	Time Since Last Reset LSB
0x75	MissionTime1	Time Since Last Reset Low Middle Byte
0x76	MissionTime2	Time Since Last Reset High Middle Byte
0x77	MissionTime3	Time Since Last Reset MSB

8.2.1 BootCount

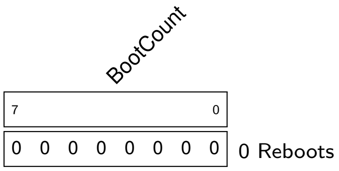
Address: 0x60

Data Type: uint8_t

Description: The BootCount register contains the number of times the Elysium microcontroller has rebooted since initial programming.

Diagram:

Register 8.43: BootCount (0x60)



Fields:

- BootCount - Number of reboots - 0x60

Notes: This register will be reset to 0 anytime a new firmware image is loaded unless a custom firmware image is built to set a different value.

After initial programming, every reboot, whether expected or unexpected, increases the value of this register by 1.

8.2.2 GPOState

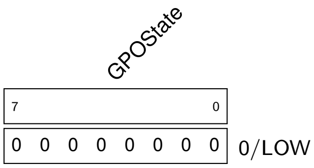
Address: 0x61

Data Type: uint8_t

Description: The GPIOState register contains the current state of the General Purpose Output pin.

Diagram:

Register 8.44: GPOSTATE (0x61)



Fields:

- GPOState - State of the GPO pin. - 0x61

Notes: None.

8.2.3 CRITErrors

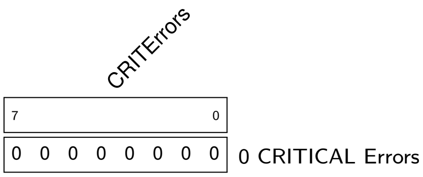
Address: 0x62

Data Type: uint8_t

Description: The CRITErrors register contains the number of CRITICAL-priority errors which have occurred since initial programming.

Diagram:

Register 8.45: CRITErrors (0x62)



Fields:

- CRITErrors - Number of CRITICAL errors - 0x62

Notes: None.

8.2.4 ERRErrors

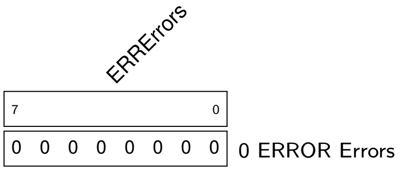
Address: 0x63

Data Type: uint8_t

Description: The ERRErrors register contains the number of ERROR-priority errors which have occurred since initial programming.

Diagram:

Register 8.46: ERRErrors (0x63)



Fields:

- ERRErrors - Number of ERROR errors - 0x63

Notes: None.

8.2.5 WARNErrors

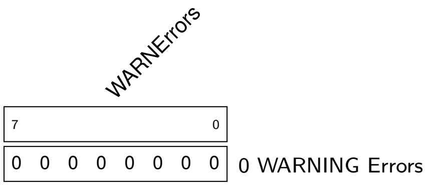
Address: 0x64

Data Type: uint8_t

Description: The WARNErrors register contains the number of WARNING-priority errors which have occurred since initial programming.

Diagram:

Register 8.47: WARNErrors (0x64)



Fields:

- WARNErrors - Number of WARNING errors - 0x64

Notes: None.

8.2.6 INFOErrors

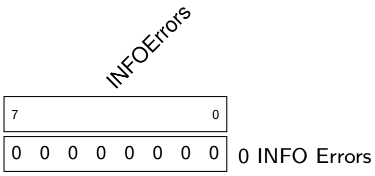
Address: 0x65

Data Type: uint8_t

Description: The INFOErrors register contains the number of INFO-priority errors which have occurred since initial programming.

Diagram:

Register 8.48: INFOErrors (0x65)



Fields:

- INFOErrors - Number of INFO errors - 0x65

Notes: None.

8.2.7 DEBUGErrors

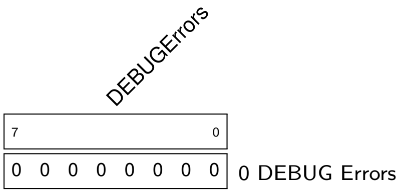
Address: 0x66

Data Type: uint8_t

Description: The DEBUGErrors register contains the number of DEBUG-priority errors which have occurred since initial programming.

Diagram:

Register 8.49: DEBUGERRORS (0x66)



Fields:

- DEBUGErrors - Number of DEBUG errors - 0x66

Notes: None.

8.2.8 LastError

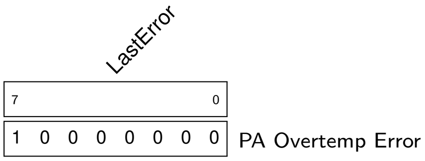
Address: 0x67

Data Type: uint8_t

Description: The LastError register contains the ID of the most recent error which has occurred.

Diagram:

Register 8.50: LASTERROR (0x67)



Fields:

- LastError - ID of most recent error - 0x67

Notes: None.

8.2.9 LastEvent

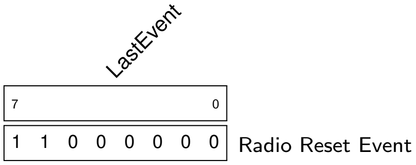
Address: 0x68

Data Type: uint8_t

Description: The LastError register contains the ID of the most recent error which has occurred.

Diagram:

Register 8.51: LastEvent (0x68)



Fields:

- LastEvent - ID of most recent event - 0x68

Notes: None.

8.2.10 ActiveBank

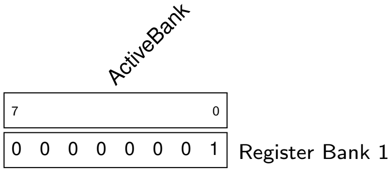
Address: 0x69

Data Type: uint8_t

Description: The ActiveBank register contains the index of the Register Bank from which the current configuration has been loaded.

Diagram:

Register 8.52: ActiveBank (0x69)



Fields:

- ActiveBank - Index of most recently loaded Register Bank - 0x69

Notes: None.

8.2.11 TelemIndex

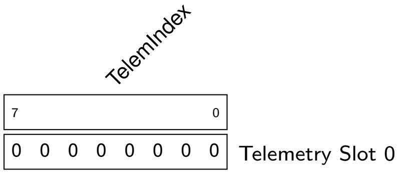
Address: 0x6A

Data Type: uint8_t

Description: The TelemIndex register contains the index of the Telemetry slot into which the next Telemetry item will be stored.

Diagram:

Register 8.53: TELEMINDEX (0x6A)



Fields:

- TelemIndex - Index of next Telemetry Slot - 0x6A

Notes: None.

8.2.12 Padding

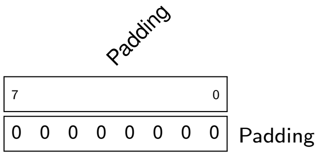
Address: 0x6B

Data Type: N/A

Description: A non-functional register used to align the Uptime and Mission Time registers on 4-byte boundaries.

Diagram:

Register 8.54: PADDING (0x6B)



Fields:

- Padding - Unused padding data - 0x6B

Notes: None.

8.2.13 Uptime[0-3]

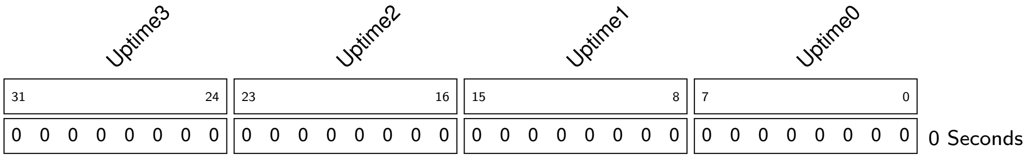
Address: 0x6C

Data Type: uint32_t (s)

Description: The Uptime register contains the number of seconds since the most recent reboot of the Elysium.

Diagram:

Register 8.55: UPTIME (0x6C)



Fields:

- Uptime3 - MSB - 0x6F
- Uptime2 - High Middle Byte - 0x6E
- Uptime1 - Low Middle Byte - 0x6D
- Uptime0 - LSB - 0x6C

Notes: This counter is reset after every reboot, expected or unexpected.

8.2.14 RFConfig[0-1]

Address: 0x70

Data Type: Bitfields

Description: The RFConfig register contains a number of bitfields defining the physical configuration of the RF sections of the Elysium.

Diagram:

Register 8.56: RFCONFIG (0x70)

RX Band ID	TX Band ID	Res.	LNA Present	Max Output Power
15 12	11 8	7 4	3	2 0
0 0 0 0	0 0 0 0	0 0 0 0	0	0 0 1

Fields:

- RX Band ID - Value identifying RX tuning band - 0x71.4
- TX Band ID - Value identifying TX tuning band - 0x71.0
- Res. - Reserved. These bits are ignored. - 0x70.4
- LNA Present - Bit identifying LNA bias supply - 0x70.3
- Max Output Power - Maximum RF output power, in Watts - 0x70.0

Notes: The reserved bit is ignored and may safely be set to any value.

The RX Band ID is a unique identifier for a particular receiver band. Currently allocated ID values are as shown below. All other values are currently reserved for future allocation.

0000 - 431.5 to 436.5 MHz

0001 - 860.5 to 877.5 MHz

0010 - 902 to 928 MHz

The TX Band ID is a unique identifier for a particular transmitter band. Currently allocated ID values are as shown below. All other values are currently reserved for future allocation.

0000 - 137 to 175 MHz

0001 - 433 to 470 MHz

0010 - 865 to 955 MHz

The LNA Present bit indicates the presence or absence of an external LNA, and thus the presence or absence of the $3.0\,\mathrm{V}$ nominal bias voltage on the RX SMA connector.

The Max Output Power bitfield contains the maximum output power possible with the installed hardware configuration, in Watts. Valid values are between 1 and 5 Watts.

8.2.15 NetworkLayer

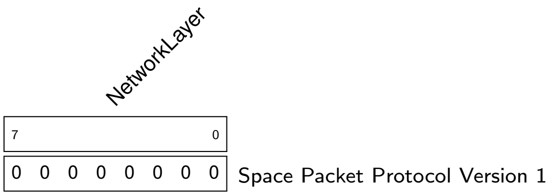
Address: 0x72

Data Type: uint8_t

Description: The NetworkLayer register contains a unique ID value identifying the specific Network Layer in use.

Diagram:

Register 8.57: NETWORKLAYER (0x72)



Fields:

- NetworkLayer - Unique ID of installed Network Layer - 0x72

Notes: The Network Layer ID uniquely identifies not only the network layer protocol but also the version of the installed Network Layer. Currently allocated ID values are as shown below. All other values are currently reserved for future allocation.

00000000 - Space Packet Protocol Version 1

00000001 - Serial Line Internet Protocol Version 1

8.2.16 DataLinkLayer

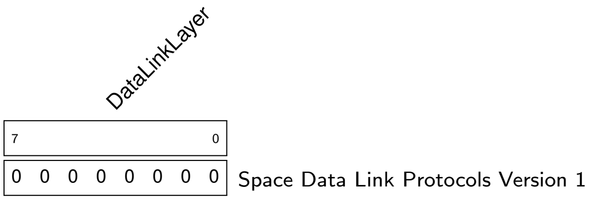
Address: 0x73

Data Type: uint8_t

Description: The DataLinkLayer register contains a unique ID value identifying the specific Network Layer in use.

Diagram:

Register 8.58: DATALINKLAYER (0x73)



Fields:

- DataLinkLayer - Unique ID of installed Data Link Layer - 0x73

Notes: The Data Link Layer ID uniquely identifies not only the data link layer protocol but also the version of the installed Data Link Layer. Currently allocated ID values are as shown below. All other values are currently reserved for future allocation.

00000000 - Space Data Link Protocols Version 1

8.2.17 MissionTime[0-3]

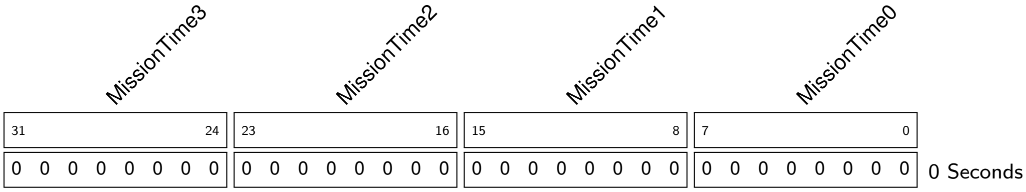
Address: 0x74

Data Type: uint32_t (s)

Description: The MissionTime register contains the number of seconds since the epoch assigned by the SetTime command.

Diagram:

Register 8.59: MissionTime (0x74)



MissionTime3	MissionTime2	MissionTime1	MissionTime0	
31 – 24	23 – 16	15 – 8	7 – 0	
0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0	0 Seconds

Fields:

- MissionTime3 - MSB - 0x77
- MissionTime2 - High Middle Byte - 0x76
- MissionTime1 - Low Middle Byte - 0x75
- MissionTime0 - LSB - 0x74

Notes: This counter is not reset on reboot.

9 On Board Telemetry Channels

The telemetry values generated by the Elysium itself are given the collective name of Channels, to distinguish them from the Telemetry which is managed by the Store Telemetry and Retreive Telemetry commands as well as certain logging facilities. The intersection of Channels and Telemetry, where Channel data is periodically logged to on board storage and thus becomes Telemetry, is discussed in the Log Channels command documentation.

The Elysium is capable of producing data from a large number of on board Channels. Some of these Channels correspond to physical sensors, such as thermistors or voltage sensors, while others correspond to more abstract notions, such as packet error rate or mission elapsed time. All Channels are treated in the same way, regardless of their source.

In addition to a name, each Channel has a unique ID code which identifies it and allows software to determine the type of data value it produces. Channel IDs are drawn from the same 8-bit space as Command, Error, and Event IDs. The 2 most significant bits of all Channel IDs are 01 - i.e., Channel IDs lie within the range 0x40-0x7F. Within this range, Channel IDs for the primary firmware lie within the range 0x40-0x5F.

Each Channel produces exactly one type of data.

A list of the Channels, their IDs, and the data type produced can be found in Table 12. A more detailed description of each Channel, including any information needed to interpret the data produced, can be found in Section 9.2.

9.1 Interacting with Channels

There are two primary ways for Channel data to be useful to the rest of the space mission.

Compute elements on board the spacecraft can subscribe to Channel data using the Subscribe to Channels command to retrieve channel data on a periodic basis (or use the Get Channel Value command to retrieve the instantaneous value of a channel). In this case, the Channel's value is sent to its destination as a standard Reply packet (as described in Section 7.2). If multiple Channel values are to be sent out at once, they are packed together into a single Reply packet by concatenation if possible. If too many Channel values are scheduled to be sent at once for them to fit into a single Reply packet, multiple Reply packets are used.

Channel data can also be logged to the on board telemetry storage using the Log Channels command. In this case, the Channel ID is prepended to the Channel's value before being packed into a Telemetry slot as described in the documentation of the Store Telemetry command.

9.2 List of Channels

A list of Channels can be found in Table 12.

Table 12: Channels

ID	*Name*	*Data Type*
0x40	RSSI	uint8_t
0x41	Output Power	uint16_t
0x42	Uplink Frame Count	uint32_t
0x43	Downlink Frame Count	uint32_t
0x44	Uplink Packet Count	uint32_t
0x45	Downlink Packet Count	uint32_t

Table 12: Channels

ID	*Name*	*Data Type*
0x46	Radio RX Bytes Count	uint32_t
0x47	Radio TX Bytes Count	uint32_t
0x48	UART RX Bytes Count	uint32_t
0x49	UART TX Bytes Count	uint32_t
0x4A	Command Received Count	uint32_t
0x4B	Reply Sent Count	uint32_t
0x4C	Microcontroller Temperature	int8_t
0x4D	Power Amplifier Temperature	int8_t
0x4E	Heat Sink Temperature	int8_t
0x4F	Last Ground Contact	int32_t
0x50	Last Spacecraft Contact	int32_t

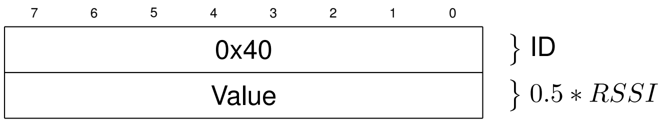
9.2.1 RSSI

Channel ID: 0x40

Data Type: uint8_t

Description: The RSSI Channel reports the Received Signal Strength Indicator (RSSI) of the receiver in units of $0.5\,\mathrm{dBm/bit}$ (i.e., it is approximately the received signal strength in dBm divided by 2).

Format:



Notes: This value is proportional to the input signal strength between approximately $-100\,\mathrm{dBm}$ and $-45\,\mathrm{dBm}$ without the LNA and between $-115\,\mathrm{dBm}$ and $-60\,\mathrm{dBm}$ when using the LNA. The value is accurate to within about $\pm 3\,\mathrm{dB}$. Better accuracy can be achieved through calibration.

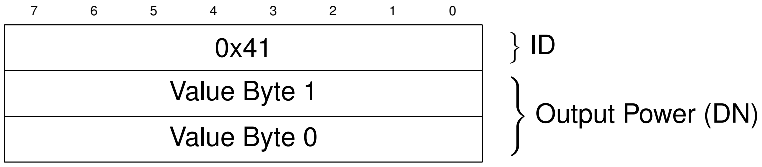
9.2.2 Output Power

Channel ID: 0x41

Data Type: uint16_t

Description: The Output Power Channel reports a digital value proportional to the actual output power of the transmitter (rather than the commanded output power).

Format:



Notes: This value is reported as a Digital Number (DN) rather than a value in any physical units. It is proportional to Output Power. For a measurement in a physical unit, a calibration should be performed.

The output power measurement drifts over temperature. For an accurate value, the calibration should be performed by correlation with the PA temperature sensor Channel.

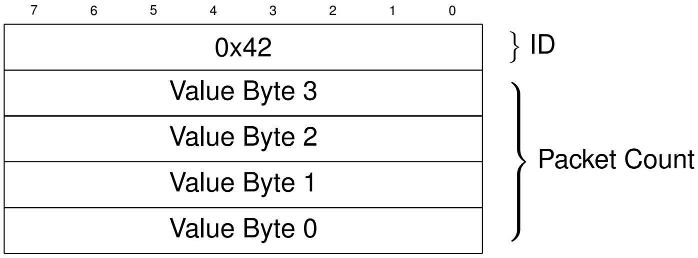
9.2.3 Uplink Frame Count

Channel ID: 0x42

Data Type: uint32_t

Description: The Uplink Frame Count Channel reports the number of frames which have been received by the radio, regardless of any error coding or frame integrity checks at the Data Link Layer.

Format:



Notes: Frames are the working units of the Data Link Layer.

This counter is increased regardless of whether the packet is later discarded due to a failure to decode or failure oof an FCS check at the Data Link Layer.

This counter increases monotonically until it rolls over to 0.

9.2.4 Downlink Frame Count

Channel ID: 0x43

Data Type: uint32_t

Description: The Downlink Frame Count Channel reports the number of frames which have been transmitted by the radio.

Format:

7 6 5 4 3 2 1 0	
0x43	ID
Value Byte 3	Packet Count
Value Byte 2	
Value Byte 1	
Value Byte 0	

Notes: Frames are the working units of the Data Link Layer.

This counter increases monotonically until it rolls over to 0.

9.2.5 Uplink Packet Count

Channel ID: 0x44

Data Type: uint32_t

Description: The Uplink Packet Count Channel reports the number of packets which have been received by the radio, regardless of any packet integrity checks at the Network Layer.

Format:

7 6 5 4 3 2 1 0	
0x44	ID
Value Byte 3	Packet Count
Value Byte 2	
Value Byte 1	
Value Byte 0	

Notes: Packets are the working units of the Networking layer.

This counter is increased regardless of whether the packet is later discarded due to a failure of an FCS check at the Networking layer.

This counter increases monotonically until it rolls over to 0.

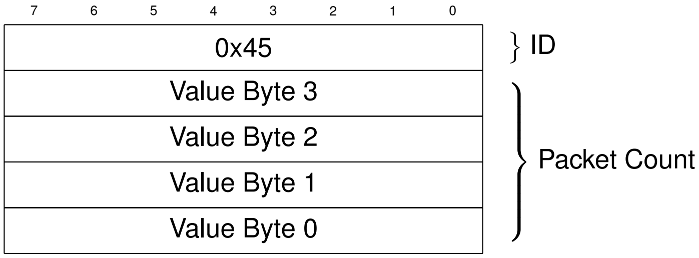
9.2.6 Downlink Packet Count

Channel ID: 0x45

Data Type: uint32_t

Description: The Downlink Packet Count Channel reports the number of packets which have been transmitted by the radio.

Format:



Notes: Packets are the working units of the Networking layer.

This counter increases monotonically until it rolls over to 0.

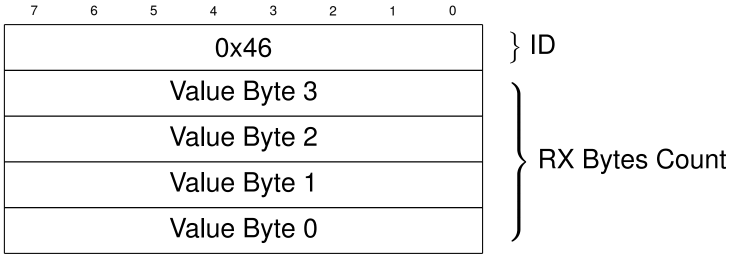
9.2.7 Radio RX Bytes Count

Channel ID: 0x46

Data Type: uint32_t

Description: The Radio RX Bytes Count Channel reports the number of bytes which have been received by the radio at the lowest level.

Format:



Notes: This counter is increased by every byte, regardless of whether it constitutes framing overhead or payload data.

This counter increases monotonically until it rolls over to 0. However, it may not increase at the instant of reception, but rather after a frame has been fully received or discarded.

9.2.8 Radio TX Bytes Count

Channel ID: 0x47

Data Type: uint32_t

Description: The Radio TX Bytes Count Channel reports the number of bytes which have been transmitted by the radio at the lowest level.

Format:

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th><th></th></tr>
<tr><td colspan="8">0x47</td><td>ID</td></tr>
<tr><td colspan="8">Value Byte 3</td><td rowspan="4">TX Bytes Count</td></tr>
<tr><td colspan="8">Value Byte 2</td></tr>
<tr><td colspan="8">Value Byte 1</td></tr>
<tr><td colspan="8">Value Byte 0</td></tr>
</table>

Notes: This counter is increased by every byte, regardless of whether it constitutes framing overhead or payload data.

This counter increases monotonically until it rolls over to 0. However, it may not increase at the instant of transmission, but rather after a frame has been fully transmitted.

9.2.9 UART RX Bytes Count

Channel ID: 0x48

Data Type: uint32_t

Description: The UART RX Bytes Count Channel reports the number of bytes which have been received over the UART interface at the lowest level.

Format:

7 6 5 4 3 2 1 0	
0x48	ID
Value Byte 3	RX Bytes Count
Value Byte 2	
Value Byte 1	
Value Byte 0	

Notes: This counter is increased by every byte, regardless of whether it constitutes framing overhead or payload data, and regardless of whether it exhibits a parity error.

This counter increases monotonically until it rolls over to 0.

9.2.10 UART TX Bytes Count

Channel ID: 0x49

Data Type: uint32_t

Description: The UART TX Bytes Count Channel reports the number of bytes which have been transmitted over the UART interface at the lowest level.

Format:

7 6 5 4 3 2 1 0	
0x49	ID
Value Byte 3	TX Bytes Count
Value Byte 2	
Value Byte 1	
Value Byte 0	

Notes: This counter is increased by every byte, regardless of whether it constitutes framing overhead or payload data.

This counter increases monotonically until it rolls over to 0.

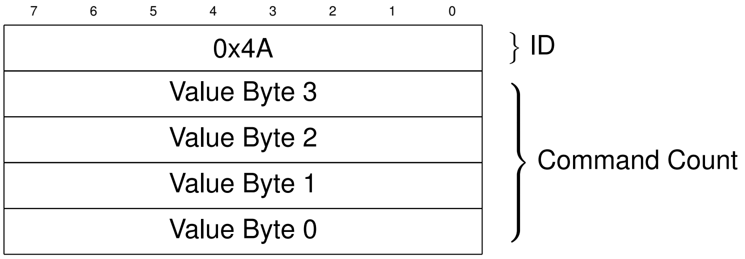
9.2.11 Command Received Count

Channel ID: 0x4A

Data Type: uint32_t

Description: The Command Received Count Channel reports the number of Elysium commands which have been received by the radio, regardless of integrity check results.

Format:



Notes: This counter is increased regardless of whether the command is later rejected due to the failure of an integrity check (e.g., FCS).

This counter increases monotonically until it rolls over to 0.

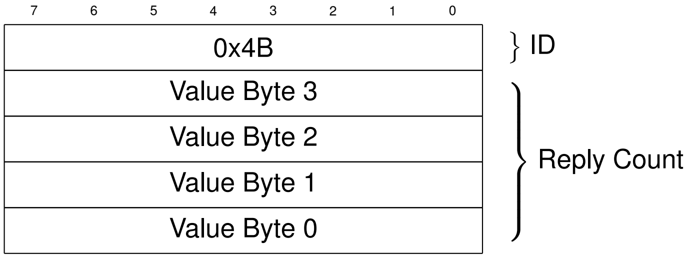
9.2.12 Reply Sent Count

Channel ID: 0x4B

Data Type: uint32_t

Description: The Reply Sent Count Channel reports the number of Reply packets which have been sent by the radio, including scheduled Channel reports.

Format:



Notes: This counter increases monotonically until it rolls over to 0.

9.2.13 Microcontroller Temperature

Channel ID: 0x4C

Data Type: int8_t

Description: The Microcontroller Temperature Channel reports the current temperature of the on board microcontroller (MCU) in degrees Celsius.

Format:

7	6	5	4	3	2	1	0	
9x72								} ID
Value								} Temp (C)

Notes: The MCU temperature is accurate to within $\pm 3\,^{\circ}\mathrm{C}$ from $30\,^{\circ}\mathrm{C}$ to $85\,^{\circ}\mathrm{C}$ and $\pm 10\,^{\circ}\mathrm{C}$ from $-40\,^{\circ}\mathrm{C}$ to $85\,^{\circ}\mathrm{C}$ (typical values are much better). A more accurate value can be achieved through further calibration.

The MCU temperature sensor is quite linear - the bulk of the error can be calibrated out with a first-order calibration (offset correction and gain error correction).

9.2.14 Power Amplifier Temperature

Channel ID: 0x4D

Data Type: int8_t

Description: The Power Amplifier Temperature Channel reports the current temperature of the power amplifier in degrees Celsius, measured by thermocouple RT1 as shown in Figure 8.

Format:

7	6	5	4	3	2	1	0	
0x4D								} ID
Value								} Temp (C)

Notes: The PA temperature measurement is accurate to within $\pm 2\,^{\circ}\mathrm{C}$ from $0\,^{\circ}\mathrm{C}$ to $85\,^{\circ}\mathrm{C}$ and $\pm 5\,^{\circ}\mathrm{C}$ from $-40\,^{\circ}\mathrm{C}$ to $85\,\mathrm{celsius}$. A more accurate value can be achieved through further calibration.

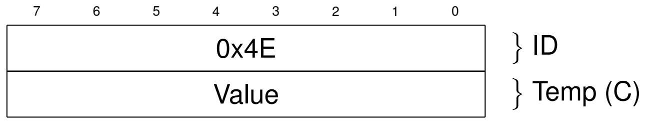
9.2.15 Heat Sink Temperature

Channel ID: 0x4E

Data Type: int8_t

Description: The Heat Sink Temperature Channel reports the current temperature of the heat sink in degrees Celsius, measured by thermocouple RT2 as shown in Figure 8.

Format:



Notes: The heat sink temperature measurement is accurate to within $\pm 2\,^{\circ}\mathrm{C}$ from $0\,^{\circ}\mathrm{C}$ to $85\,^{\circ}\mathrm{C}$ and $\pm 5\,^{\circ}\mathrm{C}$ from $-40\,^{\circ}\mathrm{C}$ to $85\,\mathrm{celsius}$. A more accurate value can be achieved through further calibration.

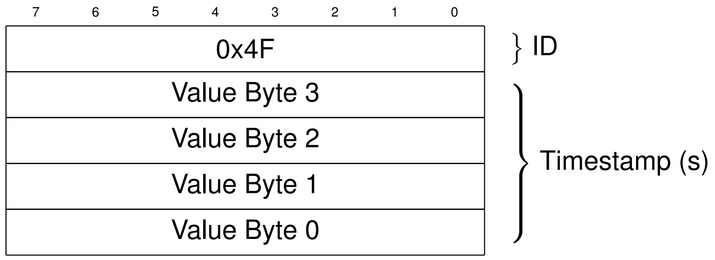
9.2.16 Last Ground Contact

Channel ID: 0x4F

Data Type: int32_t

Description: The Last Ground Contact Channel reports the timestamp of the last frame successfully received by the radio from the ground in Mission Time.

Format:



Notes: This timestamp is updated only on successful reception of a frame - that is, a frame which is successfully decoded by FEC (if present) and which passes its FCS check (if enabled) at the Data Link and Network layers.

9.2.17 Last Spacecraft Contact

Channel ID: 0x50

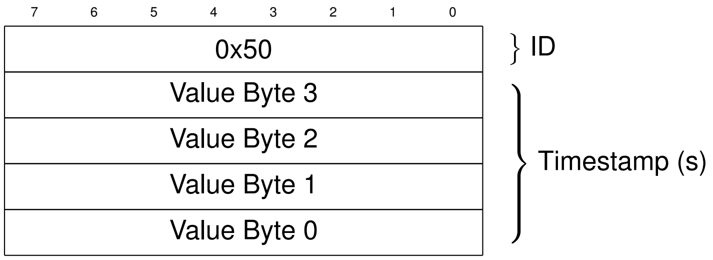
Data Type: int32_t

Description: The Last Spacecraft Contact Channel reports the timestamp of the last frame successfully received by the Elysium from the spacecraft over the UART link in Mission Time.

Format:

Table 13: Error Level Encoding

Level	*Value*
CRITICAL	0x10
ERROR	0x08
WARNING	0x04
INFO	0x02
DEBUG	0x01



Notes: This timestamp is updated only on successful reception of a frame - that is, a frame which passes its FCS check (if enabled) at the Data Link and Network layers.

10 Errors

While the Elysium is designed to be as resistant to errors as possible, anomalies do sometimes occur. This chapter will discuss the types of errors which may be reported by the radio and the autonomous fault responses which exist to handle them.

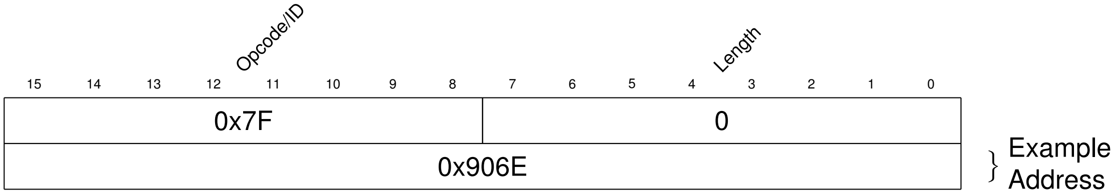
Errors are only generated if enabled in the ErrRptLvl register. After generation, errors are filtered for reporting or logging as described in the next section.

10.1 Error Reporting

Many different things may constitute an error - some examples include a sensor value exceeding a programmed threshold, a packet integrity check failing, and a spurious reset. Each error has an associated priority, often defined in a register. The list of priorities can be found in Table 13.

A mask, stored in the ErrorReportLvl register, controls which priority levels generate Error messages. Error messages are sent to an address specified in the ErrorReportAddr register

Error messages have the same format as Reply messages, as defined in Section 7.2 - the only difference is that the Opcode/ID field contains the Error ID rather than the Opcode of a command. Error messages have no Data payload. The FCS is not used for Error messages. An example of a PA Temperature Error message is below:



A separate mask, stored in the ErrorLogLvl register, controls which priority levels will be logged into on board storage as Telemetry. Only the Error ID is logged. Error reports are packed together by concatenation with other logged data produced during the same second in the same way as Channel data or other Events.

Error IDs are drawn from the same 8-bit space as Command, Channel, and Event IDs. The 2 most significant bits of Error IDs are 10 - i.e., Error IDs lie within the range 0x80-0xBF. Within this range, Error IDs for the primary firmware lie within the range 0x80-0x9F.

A list of the Errors the Elysium reports can be found in Table 14. Detailed descriptions of individual Errors can be found in the sections below.

10.2 List of Errors

Table 14: Errors

ID	*Error*
0x40	PA Overtemp
0x41	Heat Sink Overtemp
0x42	MCU Overtemp
0x43	Ground Comm Fault
0x44	Spacecraft Comm Fault
0x45	LNA Overcurrent
0x46	PA Overcurrent
0x47	MCU Undervoltage
0x48	Register Value Clipping
0x49	Invalid Opcode
0x4A	Invalid Length
0x4B	FCS Error
0x4C	Unreported Command Failure
0x4D	Subscription Overwritten
0x4E	Spurious Reset
0x4F	UART Error

10.2.1 PA Overtemp

Error ID: 0x40

Description: The PA Overtemp error indicates that the power amplifier temperature, as measured by thermistor RT1 (see Figure 8), has exceeded the value set in either the the PATempWARN register or the PATempERR register.

Generally this means that the amplifier is dissipating more heat than the heat sink is able to handle. Assuming that the thresholds have been set correctly, this indicates a potentially damaging condition which must be corrected as quickly as possible.

Fault Response? Yes - see Section 10.3.1.

Recommended Priority: N/A (fixed)

Priority Register: Fixed - WARNING and ERROR

10.2.2 Heat Sink Overtemp

Error ID: 0x41

Description: The Heat Sink Overtemp error indicates that the heatsink temperature, as measured by thermistor RT2 (see Figure 8), has exceeded the value set in either the the HSTempWARN register or the HSTempERR register.

Generally this means that the amplifier is dissipating more heat than the heat sink is able to handle. Assuming that the thresholds have been set correctly, this indicates a potentially damaging condition which must be corrected as quickly as possible.

Fault Response? Yes - see Section 10.3.1.

Recommended Priority: N/A (fixed)

Priority Register: Fixed - WARNING and ERROR

10.2.3 MCU Overtemp

Error ID: 0x42

Description: The MCU Overtemp error indicates that the microcontroller temperature, as measured by the on board thermal diode, has exceeded the value set in either the the BoardTempWARN register or the BoardTempERR register.

Generally this indicates a short circuit or overvoltage, which may be caused by a radiation event or other anomaly. Assuming that the thresholds have been set correctly, this indicates a potentially damaging condition which must be corrected as quickly as possible, perhaps by removing power to the Elysium briefly in an attempt to clear the fault condition.

This condition may also occur if the power amplifier is dissipating its heat into the board rather than the heat sink. This is also potentially damaging but may not be correctable in an on orbit asset.

Fault Response? Yes - see Section 10.3.1.

Recommended Priority: N/A (fixed)

Priority Register: Fixed - WARNING and ERROR

10.2.4 Ground Comm Fault

Error ID: 0x43

Description: The Ground Comm Fault error indicates that no packets have been received by the radio from the ground for a duration set by the GFTime register. This can be caused by a configuration error or by hardware damage.

Fault Response? Yes - see Section 10.3.2.

Recommended Priority: ERROR

Priority Register: GFLvl

10.2.5 Spacecraft Comm Fault

Error ID: 0x44

Description: The Spacecraft Comm Fault error indicates that no data has been received by the radio from the rest of the spacecraft over the UART for a duration set by the SCCommTime register. This can be caused by many things, including configuration errors and hardware damage.

Fault Response? Yes - see Section 10.3.3.

Recommended Priority: WARNING

Priority Register: SCCommLvl

10.2.6 LNA Overcurrent

Error ID: 0x45

Description: The LNA Overcurrent error indicates that the off board LNA being powered by the on board bias tee is drawing more than $300\,\mathrm{mA}$ of current. This indicates a short circuit on the board, potentially a latch-up caused by a radiation event. This is an **extremely** damaging condition and must be corrected as quickly as possible. For this reason, the fault response for this condition is non-optional and cannot be disabled.

Fault Response? Yes - see Section 10.3.4.

Recommended Priority: N/A (fixed)

Priority Register: Fixed - CRITICAL

10.2.7 PA Overcurrent

Error ID: 0x46

Description: The PA Overcurrent error indicates that the power amplifier is drawing more than $3\,\mathrm{A}$ of current. This indicates either a short circuit on the board, potentially a latch-up caused by a radiation event, or a serious overdrive of the power amplifier. Either of these represents an **extremely** damaging condition and must be corrected as quickly as possible. For this reason, the fault response for this condition is non-optional and cannot be disabled.

Fault Response? Yes - see Section 10.3.4.

Recommended Priority: N/A (fixed)

Priority Register: Fixed - CRITICAL

10.2.8 MCU Undervoltage

Error ID: 0x47

Description: The MCU Undervoltage error indicates that the microcontroller is receiving less than its expected $2.375\,\mathrm{V}$ minimum voltage. This is typically the result of a short circuit, potentially caused by a radiation event, although it may also be caused by insufficient voltage at the $5\,\mathrm{V}$ input pin. While the undervoltage itself is unlikely to cause hardware damage, it may cause memory corruption. In addition, if the undervoltage is caused by a short circuit, that may be causing hardware damage. It is recommended that this condition be corrected as quickly as possible, perhaps by removing power from the Elysium for a short time in an attempt to clear the fault.

Fault Response? No.

Recommended Priority: N/A (fixed)

Priority Register: Fixed - CRITICAL

10.2.9 Register Value Clipping

Error ID: 0x48

Description: The Register Value Clipping error indicates that a Set Registers or Set Block command was sent which contained an invalid register value. The invalid register value was clipped to the nearest valid value.

Fault Response? No.

Recommended Priority: WARNING

Priority Register: RegErrLvl

10.2.10 Invalid Opcode

Error ID: 0x49

Description: The Invalid Opcode error indicates that a command packet was received with an invalid value in its Opcode field. The packet was discarded and no action was taken.

Fault Response? No.

Recommended Priority: WARNING

Priority Register: OpErrLvl

10.2.11 Invalid Length

Error ID: 0x4A

Description: The Invalid Length error indicates that a command packet was received with a value in its Length field which did not match any of the possible values expected for the command specified by the Opcode field, or which exceeded 251 bytes. The packet was discarded and no action was taken.

Fault Response? No.

Recommended Priority: WARNING

Priority Register: LengthErrLvl

10.2.12 FCS Error

Error ID: 0x4B

Description: The FCS Error error indicates that a command packet was received with a value in its FCS field which did not match the calculated CRC value derived from the rest of the packet. The packet was discarded and no action was taken.

Fault Response? No.

Recommended Priority: WARNING

Priority Register: FCSLvl

10.2.13 Unreported Command Failure

Error ID: 0x4C

Description: The Unreported Command Failure error indicates that a command packet was received with an invalid data value for the command specified by the Opcode field, and further that the Reply Requested bit was not set (and therefore no FAILURE message was sent out in response). The packet was discarded and no action was taken.

Fault Response? No.

Recommended Priority: WARNING

Priority Register: NRErrLvl

10.2.14 Subscription Overwritten

Error ID: 0x4D

Description: The Subscription Overwritten error indicates that a subscription command (Subscribe to Channels or Subscribe to Events) was received which requested a subscription to a channel or event which had already been subscribed to. The new subscription information overwrote the older information.

Fault Response? No.

Recommended Priority: INFO

Priority Register: SubOverLvl

10.2.15 Spurious Reset

Error ID: 0x4E

Description: The Spurious Reset error indicates that the Elysium detected that it had been reset, but that it had not received a Reset or Install Firmware command. This may happen due to the spacecraft removing power from the Elysium temporarily, or due to many other possible events, some benign and some dangerous.

Fault Response? No.

Recommended Priority: ERROR

Priority Register: ResetErrLvl

10.2.16 UART Error

Error ID: 0x4F

Description: The UART Error error indicates that a framing or parity error occurred on the UART interface.

Fault Response? No.

Recommended Priority: WARNING

Priority Register: UARTErrLvl

10.3 Fault Handlers

Certain errors represent dangers to the Elysium hardware or to the rest of the spacecraft. These errors have autonomous fault handlers in place to mitigate the potential damage. Some of these fault handlers may be disabled through register settings, while others are critical enough that their use is non-optional.

Details of the Elysium's fault handers can be found in the sections below.

10.3.1 Overtemp Fault Response

The Elysium is capable of detecting three different overtemp conditions - the PA, the heat sink, and the MCU. One or more of these conditions may be configured in the FaultResponse register to trigger this fault response.

The action taken is the same regardless of which overtemp condition triggers the fault response. The transmitter is stopped and power is removed from the PA for a duration specified in the OTFaultTime register. The transmit queue is maintained intact and data begins transmitting again at the end of the specified interval.

If the overtemperature condition has not cleared at the end of the interval, this fault response will be triggered again immediately.

10.3.2 Ground Comm Lost Fault Response

When the Elysium has been out of contact with the ground station for the amount of time specified in the GFTime register, this fault respnse may be triggered if it is enabled in the FaultResponse register.

This fault response has two modes, either or both of which may be enabled in the FaultResponse register. If the GroundFaultFallback bit is set, the Elysium will reload its configuration from the Register Bank specified in the GFBank register. If this fault response is in use, it is advisable to define a “failsafe” configuration for the Elysium and maintain a Register Bank with that configuration, then ensure the value of the GFBank register in all other Register Banks points to the failsafe Register Bank.

The ground fault timer will start again after the configuration has been reloaded if enabled in the FaultResponse register of the fallback Register Bank (potentially with a new duration as specified by the GFTime register in the newly loaded Register Bank). This allows the user to chain fault responses if desired by setting the GFBank register of the newly loaded Register Bank to another Register Bank.

If the GroundFaultBeacon bit is set, the Elysium will begin broadcasting Beacon frames at the interval specified in the BeaconTime register. The exact format of Beacon frames depends on the Data Link Layer - for example, the SDLP Data Link Layer transmits Only Idle Data (OID) frames as Beacon frames. This fault response is designed to help locate the spacecraft if its position cannot be determined.

10.3.3 Spacecraft Comm Lost Fault Response

When the Elysium has been out of contact with the rest of the spacecraft for the amount of time specified in the SCCommTime register, this fault response may be triggered if it is enabled in the FaultResponse register.

This fault response has two modes, either or both of which may be enabled in the FaultResponse register. If the SCCommFaultRst bit is set, the Elysium will reset itself (potentially causing a Spurious Reset error to be reported) in an attempt to clear any problems with the UART hardware. If the SCCommFaultGPO bit is set, the Elyisum will toggle the GPO pin for $1\,\mathrm{ms}$ in an attempt to reset external hardware which may be in a hung state.

In either case, the UART baud rate stored in the SCCommRate register will be configured as the new UART baud rate, to allow fallback to a "failsafe" configuration.

The spacecraft fault timer will not start again after the reset until a byte is received on the UART interface.

10.3.4 Overcurrent Fault Responses

The Elysium can detect overcurrents on both the PA and LNA power rails. When an overcurrent is detected on a power rail, that rail is disabled for $5\,\mathrm{ms}$ in an attempt to clear the overcurrent condition. After power has been restored, the overcurrent condition will be checked again. If the overcurrent condition still exists, the rail will be disabled for a further $5\,\mathrm{ms}$. This cycle may repeat up to 5 times. After 5 consecutive overcurrent faults, the radio is considered to be irreparably damaged. The damaged rail is powered off until the next reset of the Elysium in an attempt to preserve the functionality of undamaged parts of the radio. If the fault is cleared through external action, the radio may be reset to restore full functionality.

These fault responses are non-optional due to the high risk presented by a short circuit.

11 Events

While Channels provide a convenient mechanism through which to access time-series data, some events occur infrequently or must be reacted to immediately. The Events system allows asynchronous reporting of interesting data points while being substantially similar to the Channels interface.

In addition to a name, each Event has a unique ID code which identifies it. Event packets carry no information other than the Event ID. A list of the Events and their IDs can be found in Table 15. A more detailed description of each Event can be found in Section 11.2.

Event IDs are drawn from the same 8-bit space as Command, Channel, and Error IDs. The 2 most significant bits of Event IDs are 11 - i.e., Event IDs lie within the range 0xC0-0xFF. Within that range, Event IDs for the primary firmware lie within the range 0xC0-0xDF.

11.1 Interacting with Events

There are two primary ways for Events to be useful to the rest of the space mission.

Compute elements on board the spacecraft can subscribe to Events using the Subscribe to Events command to be notified of Events asynchronously when they occur. In this case, the Event notification is sent to its destination as a standard Reply packet (as described in Section 7.2), with the Event ID in the Opcode/ID field and no data payload. Each Event gets its own Reply packet. The FCS function is not used with Events.

Event reports can also be logged to the on board telemetery storage using the Log Events command. In this case, the Event ID is packed into a Telemetry slot as described in the documentation of the Store Telemetry command.

11.2 List of Events

A list of Events can be found in Table 15

Table 15: Events

ID	*Event*
0xC0	Radio Reset
0xC1	Configuration Reloaded
0xC2	Mission Time Changed
0xC3	GPO State Changed
0xC4	Telemetry Rollover
0xC5	Transmit Frequency Changed
0xC6	Receive Frequency Changed
0xC7	Transmit Bit Rate Changed
0xC8	Receive Bit Rate Changed
0xC9	Transmit Deviation Changed
0xCA	Receive Deviation Changed
0xCB	Output Power Changed
0xCC	UART Baud Rate Changed
0xCD	Frame Received
0xCE	Frame Transmitted
0xCF	Packet Received
0xD0	Packet Transmitted
0xD1	Command Received
0xD2	Reply Sent

11.2.1 Radio Reset

Event ID: 0xC0

Description: The Radio Reset Event indicates that the Elysium radio has been reset or power cycled.

Notes: Unlike the Spurious Reset error, the Radio Reset Event makes no distinction between “planned” and “unplanned” resets.

11.2.2 Configuration Reloaded

Event ID: 0xC1

Description: The Configuration Reloaded Event indicates that the Elysium radio’s configuration has been reloaded, either as the result of a ReloadConfig command or due to a fault response.

Notes: None.

11.2.3 Mission Time Changed

Event ID: 0xC2

Description: The Mission Time Changed Event indicates that the current Mission Time has changed in a discontinuous way, either due to a SetTime command or due to rollover of the Mission Time counter.

Notes: When the Mission Time counter rolls over, it returns to $-(2^{31})$, or approximately 68 years before the epoch.

The new Mission Time can be retrieved with the GetTime command.

11.2.4 GPO State Changed

Event ID: 0xC3

Description: The GPO State Changed Event indicates that the state of the General Purpose Output pin has changed, either due to a SetGPO command or due to a fault response.

Notes: The current state of the GPO pin can be retrieved with the GetGPO command.

11.2.5 Telemetry Rollover

Event ID: 0xC4

Description: The Telemetry Rollover Event indicates that the “current telemetry slot” has wrapped around from telemetry slot 255 to telemetry slot 0.

Notes: Telemetry is overwritten based on the current telemetry slot. After this event the telemetry in low-numbered slots will begin to be overwritten.

The current telemetry slot is stored in special Bank 0 register TelemIndex.

11.2.6 Transmit Frequency Changed

Event ID: 0xC5

Description: The Transmit Frequency Changed Event indicates that the transmit frequency has been changed by a SetTXFreq command.

Notes: The new transmit frequency can be retrieved with the GetTXFreq command.

11.2.7 Receive Frequency Changed

Event ID: 0xC6

Description: The Receive Frequency Changed Event indicates that the receiver frequency has been changed by a SetRXFreq command.

Notes: The new receiver frequency can be retrieved with the GetRXFreq command.

11.2.8 Transmit Bit Rate Changed

Event ID: 0xC7

Description: The Transmit Bit Rate Changed Event indicates that the transmit bit rate has been changed by a SetTXRate command.

Notes: The new transmit bit rate can be retrieved with the GetTXRate command.

11.2.9 Receive Bit Rate Changed

Event ID: 0xC8

Description: The Receive Bit Rate Changed Event indicates that the receiver bit rate has been changed by a SetRXRate command.

Notes: The new receiver bit rate can be retrieved with the GetRXRate command.

11.2.10 Transmit Deviation Changed

Event ID: 0xC9

Description: The Transmit Deviation Changed Event indicates that the transmit deviation has been changed by a SetTXDev command.

Notes: The new transmit deviation can be retrieved with the GetTXDev command.

11.2.11 Receive Deviation Changed

Event ID: 0xCA

Description: The Receive Deviation Changed Event indicates that the receiver deviation has been changed by a SetRXDev command.

Notes: The new receiver deviation can be retrieved with the GetRXDev command.

11.2.12 Output Power Changed

Event ID: 0xCB

Description: The Output Power Changed event indicates that the commanded transmitter output power has been changed by a SetTXPow command.

Notes: This event relates to commanded power and not measured power - for measured output power see the Output Power Channel.

The new commanded power can be retrieved with the GetTXPow command.

11.2.13 UART Baud Rate Changed

Event ID: 0xCC

Description: The UART Baud Rate Changed Event indicates that the baud rate of the UART interface has been changed by a SetBaud command or by the Autobaud functionality.

Notes: The new baud rate can be retrieved with the GetBaud command.

11.2.14 Frame Received

Event ID: 0xCD

Description: The Frame Received Event indicates that an uplink frame has been received over the radio interface.

Notes: The definition of a frame is handled by the Data Link Layer.

This event is reported regardless of the success or failure of any frame integrity checks or coding that may be performed at the Data Link Layer.

The total number of received frames can be found in the Uplink Frame Count Channel.

11.2.15 Frame Transmitted

Event ID: 0xCE

Description: The Frame Transmitted Event indicates that a downlink frame has been sent over the radio interface.

Notes: The definition of a frame is handled by the Data Link Layer.

The total number of transmitted frames can be found in the Downlink Frame Count Channel.

11.2.16 Packet Received

Event ID: 0xCF

Description: The Packet Received Event indicates that an uplink packet has been received over the radio interface.

Notes: The definition of a packet is handled by the Networking Layer.

This event is reported regardless of the success or failure of any packet integrity checks that may be performed at the Networking Layer.

The total number of received packets can be found in the Uplink Packet Count Channel.

11.2.17 Packet Transmitted

Event ID: 0xD0

Description: The Packet Transmitted Event indicates that a downlink packet has been transmitted over the radio interface.

Notes: The definition of a packet is handled by the Networking Layer.

The total number of transmitted packets can be found in the Downlink Packet Count Channel.

11.2.18 Command Received

Event ID: 0xD1

Description: The Command Received Event indicates that an Elysium command packet has been received by the firmware.

Notes: This event is reported regardless of the success or failure of any of the command packet integrity checks.

The total number of received commands can be found in the Command Received Count Channel.

11.2.19 Reply Sent

Event ID: 0xD2

Description: The Reply Sent Event indicates that the Elysium firmware has sent a Reply packet in response to a command.

Notes: This event is reported only for command responses, not for Channel data or Event reporting.

The total number of replies sent can be found in the Reply Sent Count channel.

A References

- Isola Group FR408 High-Speed Epoxy Laminate <http://www.isola-group.com/products/fr408/ >.
- Panasonic ERT-J0EP473F Thermistor <http://industrial.panasonic.com/ww/products/thermal-solutions/ntc-thermistor-chip-type/ntc-thermistor/ntc-thermistorchip-type/ERTJ0EP473F >.
- Samtec TSW Series 100-mil Header Connectors <https://www.samtec.com/products/tsw >.
- Samtec SIB Series 100-mil One-Piece Compression Connectors <https://www.samtec.com/products/sib >.
- Samtec PHT Series 100-mil Press Fit Header Connectors <https://www.samtec.com/products/pht >.
- Samtec PHT Series 100-mil Press Fit Header Connectors <https://www.samtec.com/products/pht >.
- Samtec SMA Series Edge-Mount RF Connectors <https://www.samtec.com/products/sma-em >.
- OSI Basic Reference model <http://www.iso.org/iso/catalogue_detail.htm?csnumber=20269 >.
- Elysium page of the Adamant website <http://www.adamantaerospace.com/elysium-vhfuhf-radio/ >.
- CCSDS TM Space Data Link Protocol [PDF] <https://public.ccsds.org/Pubs/132x0b2.pdf >.
- CCSDS TC Space Data Link Protocol [PDF] <https://public.ccsds.org/Pubs/232x0b3.pdf >.
- ITU X.25 Specification <https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.25-199610-I!!PDF-E&type=items >.

B Revision History

1. Initial release
2. Updated draft for Revision B PCBs

3. Updated draft with register location changes