Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Engenharia Eletrônica

Sistema de Visualização de Telemetria e Envio de Telecomando para o CubeSat AlfaCrux

Autor: Priscila Yukie Yamada

Orientador: Prof. Dr. William Reis Silva

Coorientador: Prof. Dr. Renato Alves Borges

Brasília, DF

2022

Priscila Yukie Yamada

Sistema de Visualização de Telemetria e Envio de Telecomando para o CubeSat AlfaCrux

Monografia submetida ao curso de graduação em Engenharia Eletrônica da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia Eletrônica.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Prof. Dr. William Reis Silva

Coorientador: Prof. Dr. Renato Alves Borges

Brasília, DF

2022

Priscila Yukie Yamada
Sistema de Visualização de Telemetria e Envio de Telecomando para o CubeSat AlfaCrux/ Priscila Yukie Yamada. – Brasília, DF, 2022-
93 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. William Reis Silva

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2022.

1. AlfaCrux. 2. Visualização de Telemetrias. 3. Telecomando. 4. Cosmos. I. Prof. Dr. William Reis Silva. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Sistema de Visualização de Telemetria e Envio de Telecomando para o CubeSat AlfaCrux

Priscila Yukie Yamada

Sistema de Visualização de Telemetria e Envio de Telecomando para o CubeSat AlfaCrux

Monografia submetida ao curso de graduação em Engenharia Eletrônica da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia Eletrônica.

Trabalho aprovado. Brasília, DF, 04 de Maio de 2022:

Prof. Dr. William Reis Silva
Orientador

Prof. Dr. Renato Alves Borges
Coorientador

Prof. Dr. Giancarlo Santilli
Convidado 1

MSc. Eng. Wilson Yamaguti
Convidado 2

Brasília, DF
2022

Agradecimentos

A Deus, pela minha vida, minha saúde e por ter me dado força e determinação para superar os obstáculos durante a graduação.

Aos meus pais, Cláudia e Sérgio, que sempre me incentivaram e apoiaram. Em especial, a minha mãe, presente em todas as conquistas, me aconselhando e me animando para não desanimar.

Aos meus irmãos, Luciana e Thayuan, que sempre me ajudaram. Em especial, a minha irmã que me acompanhou em todo o desenvolvimento desse trabalho, me aconselhando e revisando a escrita.

A toda minha família e parentes, tios, primos, avôs, que me incentivaram a concluir o curso e no início de uma nova carreira, principalmente, no início do meu estágio na empresa Visiona Tecnologia Espacial.

A todos meus amigos, particularmente, Adriana, Fernanda, Amanda e Marina, que me acompanharam em todo o meu desenvolvimento escolar, desde o ensino fundamental, presentes com palavras de encorajamento e força.

Aos meus amigos, Larissa Barros, Larissa Martins, Mateus, Lucas, Caio, Maurício e Renato, que estiveram ao meu lado durante todo o período da graduação. Em especial, o Mateus, que desempenhou um papel significativo em meu crescimento, me aconselhando, ajudando e me dando forças para finalizar o presente trabalho.

A universidade, seu corpo docente, direção e administração, pela oportunidade de fazer o curso e proporcionar um ambiente para o crescimento intelectual com projetos de pesquisas, equipes de competições e empresa juniores.

Agradeço a todos os professores por me ensinarem o conhecimento necessário para a vida profissional, em especial meu orientador e meu coorientado, pelo apoio, confiança e orientações neste trabalho.

E ao meu orientador de estágio na empresa Visiona Tecnologia Espacial. Estimado, Wilson Yamaguti, é com muita admiração e carinho que gostaria de expressar meu agradecimento por toda paciência e dedicação de me ensinar e me orientar durante todo o desenvolvimento desse trabalho.

Resumo

As missões espaciais têm alto custo e necessitam de um longo tempo de desenvolvimento, devido a isso, foi lançado o projeto *CubeSat*, o projeto tem o propósito de ser um satélite de pequeno porte e mais simples, o que possibilitou a redução de tamanho e peso, facilitando, por exemplo, em termos de energia e custo e permitindo um acesso mais igualitário à tecnologia. Dentre as vantagens, está a oportunidade para a comunidade universitária, já que facilita a acessibilidade e possibilita lançamentos frequentes. Indo ao encontro da tendência da indústria espacial mundial, estudantes da Universidade de Brasília (UnB) iniciaram o projeto AlfaCrux, um *CubeSat* 1U radioamador, com fins educacionais e de demonstração tecnológica na área de telecomunicação. O projeto, que é um exemplo inserido dentro do meio acadêmico, tem o intuito de levar uma repetidora de pacotes digitais e serviço de armazenamento e retransmissão de dados a locais mais remotos, educando e treinando tanto os alunos quanto professores da universidade. Um ponto importante do projeto é a comunicação entre o satélite e o segmento de controle em solo através das funções de telecomando e de telemetria. Assim, um sistema de visualização de telemetria e de envio de telecomandos ou de rotinas de telecomando se tornam essenciais. Com essa importância ressaltada, o principal objetivo deste trabalho é desenvolver um sistema de visualização de dados de telemetria para o Centro de Controle de Missões do Projeto AlfaCrux e simular telecomandos para essa missão, a fim de contribuir para os avanços tecnológicos brasileiros no campo aeroespacial. A visualização dos dados e a simulação do envio de comandos serão feitos através do *framework* Cosmos, usado para controlar um conjunto de sistemas embarcados e construir telas de telemetrias personalizadas. A organização e o *layout* dessas telas são importantes para que o controlador tenha uma visualização mais intuitiva dos dados recebidos e o envio de comandos de forma correta e segura é importante para o sucesso de toda a missão.

Palavras-chaves: AlfaCrux, Visualização de Telemetrias, Telecomandos, Cosmos.

Abstract

Space missions are expensive and require a long time to develop, due to this, the CubeSat project was launched. The project's purpose is to be a small size and simpler satellite, and that weight and hight reduction easy its implementation, for example, in terms of energy and cost enabling more equal access to space technology. Among the advantages is the opportunity for the university community, as it facilitates accessibility and allows increasing launch frequency. In line with the trend of the world space industry, students at the University of Brasília (UnB) started the AlfaCrux project, a 1U CubeSat radio amateur, with educational and technological demonstration purposes in the area of telecommunications. The project, which is an example within the academic environment, aims to provide a repeater of digital packages and a data storage and forward service to more remote locations, educating and training both students and professors. An important point of the project is the communication between the satellite and the ground control segment through telecommand and telemetry functions. Thus, a system for visualizing telemetry and sending telecommands becomes essential. With this importance highlighted, the main objective of this work is to develop a telemetry data visualization system for the Mission Control Center of the AlfaCrux Project and to simulate telecommands for this mission, in order to contribute to Brazilian technological advances in the aerospace field. The data visualization and command sending simulation will be done through the Cosmos *framework*, used to control a set of embedded systems and build custom telemetry displays. The organization and *layout* of these screens are important for the controller to have a more intuitive visualization of the received data and sending commands in a correct and safe way is important for the success of the whole mission.

Key-words:AlfaCrux, Telemetry Display, Telecommand, Cosmos.

Lista de ilustrações

Lista de tabelas

Lista de quadros

Lista de abreviaturas e siglas

AEB	Agência Espacial Brasileira
API	Application Programming Interface
Cal Poly	California Polytechnic State University
CAN	Controller Area Network
CCSDS	Consultative Committee for Space Data Systems
CGEE	Centro de Gestão e Estudos Estratégicos
CRC	Cycle Redundancy Check
CSP	CubeSat Space Protocol
EPS	Electric Power System
FAPDF	Fundação de Apoio e Pesquisa do Distrito Federal
GDF	Governo do Distrito Federal
GFSK	Gaussian Frequency Shift Keying
GMSK	Gaussian Minimum Shift Keying
GPS	Global Positioning System
GRGT	Guam Remote Ground Terminal
GS	Ground Station
GSTS-ME	Ground Station System - Mission Exploitation
GSTS-SSC	Ground Station System - Space Segment Control
HDLC	High-level Data Link Control
HMAC	Hash-based Message Authentication Code
IS	Information Standards
ISO	International Standards Organization
ITU	União Internacional de Telecomunicações
I^2C	Inter-Integrated Circuit

LEO	Low Earth Orbit
LEOP	Launch and Early Orbit Phase
LODESTAR	Laboratório de Simulação e Controle de Sistemas Aeroespaciais
LSB	Least Significant Bit
MAC	Madium Access Control
MCC	Mission Control Center
MCTI	Ministério da Ciência, Tecnologia e Inovação
MSB	MostSignificant Bit
NASA	National Aeronautics and Space Administration
OBC	On-Board Computer
OBDH	On-Board Data Handling
OSI	Open Systems Interconnection
PDFF	Plano de Atribuição, Destinação e Distribuição de Frequências
P-POD	Poly Picosatellite Orbital Deployer
PUS	Packet Utilisation Standart
RF	Radio Frequency
RGB	Read, Green and Blue
RS	Reed-Solomon
SCM-αC	Sistema de Controle de Missão - AlfaCrux
SDR	Software Defined Radio
SSID	Service Set IDentifier
TDRSS	Tracking and DataRelay Satellite System
TC	Command
TCP/IP	Transmission Control Protocol/Internet Protocol
TM	Telemetry
TMTC	Telemetry and Command

TT&C	Telemetry, Tracking, and Command
UHF	Ultra High Frequency
UI-Frame	Unnumbered Information Frame
UnB	Universidade de Brasília
UnB-FGA	Universidade de Brasília - Faculdade do Gama
UnB-FT	Universidade de Brasília - Faculdade de Tecnologia
URSS	União das Repúblicas Socialistas Soviética
WSC	White Sands Complex
ZMQ	Zero-MQ

Sumário

1 INTRODUÇÃO

1.1 Contextualização e Motivação

O alcance global e a enorme carga útil de aproximadamente cinco toneladas do foguete R-7, da União das Repúblicas Socialistas Soviética (URSS) possibilitou a aprovação de lançar um satélite (CORPORATION, 2016). O lançamento em órbita terrestre do Sputnik em outubro de 1957, com o objetivo de obter conhecimento do espaço, deu início à era espacial (PELTON; MADRY; CAMACHO-LARA, 2016a).

De acordo com MCTI (2021) os satélites artificiais são usados em diversas áreas como na comunicação, observação da Terra, meteorologia, defesa e navegação, além disso têm grande importância no cotidiano da humanidade há mais de meio século, de forma a contribuir para o bem-estar e permitindo alcançar novos objetivos (CANADA, 2020).

De acordo com Pelton, Madry e Camacho-Lara (2016a), as áreas envolvendo satélites podem ser divididas em duas: satélites científicos e satélites de aplicações. Os satélites científicos exploram o mundo, o sistema solar e a galáxia. Os satélites de aplicações fornecem serviços às pessoas da Terra.

Segundo o CGEE (2018) as missões espaciais baseadas em satélites têm um alto custo e longos tempos de desenvolvimento, o que torna inviável o projeto de uma missão espacial completa no período típico de uma graduação nas áreas de Ciências Exatas e Engenharias. Nesse contexto, professores da *California Polytechnic State University* (Cal Poly) e da *University of Stanford* iniciaram o projeto *CubeSat* com o objetivo de ser um satélite de pequeno porte seguindo um padrão mais simples, com a intenção de prover a experiência de trabalho de uma missão espacial desde o início ao fim para alunos de graduação e pós-graduação (PROGRAM, 2020).

O desenvolvimento de nanossatélites tem se expandido no mercado nos últimos anos devido aos avanços tecnológicos que possibilitaram reduzir o tamanho, o peso e melhorar a eficiência em termos de energia de cargas úteis e de instrumentos para missões espaciais (TOORIAN; DIAZ; LEE, 2008). O uso de naves espaciais menores reduz o custo e o tempo de desenvolvimento e consequentemente, aumenta a acessibilidade ao espaço para fins de exploração, demonstração tecnológica e para a comunidade universitária, possibilitando sustentar lançamentos frequentes (PROGRAM, 2020; HARTWELL; LIGHTSEY, 2021).

Incluso na classe de nanossatélites estão os *CubeSats*, termo formado pela palavra cubo (do inglês, *cube*) acrescida das três primeiras letras da palavra satélite (DELMONDES, 2019). Os *CubeSats* têm estrutura, volume e massa padrão de uma unidade ou 1U medindo 10x10x10

cm e pode ser expandido para tamanhos maiores de 1,5U, 2U, 3U, 6U e até 12U (NASA, 2015).

Seguindo a indústria espacial de construção e desenvolvimento de satélites de pequeno porte, estudantes da Universidade de Brasília com o apoio da Agência Espacial Brasileira (AEB), autarquia vinculada ao Ministério da Ciência, Tecnologia e Inovações (MCTI) e da Fundação de Apoio a Pesquisa do Distrito Federal (FAPDF) iniciaram o projeto AlfaCrux. O projeto consiste em um *CubeSat* de 1U para o desenvolvimento de um sistema de comunicação de pesquisa para a sociedade civil e militar (MCTI, 2020).

De acordo com o MCTI (2020), o AlfaCrux prevê o desenvolvimento de algumas tecnologias, como: o desenvolvimento de tecnologias de controle de atitude de pequenos satélites por atuação magnética e algoritmos de determinação e controle de atitude. Dentre essas pesquisas está incluso o estudo de um sistema de visualização de telemetria e de envio de telecomandos para serem implementados em *CubeSats*.

1.2 Justificativa

O interesse na área de pequenos satélites teve um grande aumento nos últimos anos (PESQUISA-FAPESP, 2019), pois podem ser colocadas em órbita por um custo muito menor, o que permite nações não espaciais, corporações e instituições educacionais ter acesso à essas tecnologias (CGEE, 2018).

A missão AlfaCrux é um exemplo de um nanosatélite inserido dentro da universidade com o objetivo educacional e demonstração tecnológica no contexto de comunicação em banda estreita, atendendo a locais sem infra-estrutura terrestre, como áreas subdesenvolvidas, áreas desabitadas e áreas que foram devastadas por um desastre natural que conta com o apoio de professores e alunos da Universidade de Brasília, da Faculdade de Tecnologia (UnB-FT) e da Faculdade do Gama (UnB-Gama).

Cada satélite desempenha uma função em uma missão espacial, mas todos são equipados com dispositivos para enviar e receber dados (ROHLING, 2018). A comunicação de dados entre o satélite e o operador da missão é uma atividade crítica para o sucesso das missões, necessitando de uma infraestrutura de solo para receber as telemetrias e enviar os telecomandos, além de um meio para o processamento e visualização em tempo real das telemetrias (HARTWELL; LIGHTSEY, 2021).

Assim, o estudo para a criação de um sistema de processamento e visualização de telemetria de *CubeSats* se mostra de grande importância no campo de pequenos satélites, para a missão do AlfaCrux e para projetos futuros dentro das universidades, com enfoque na Universidade de Brasília.

Outrossim, o trabalho apresenta o estudo de uma ferramenta *open source* utilizada em missões espaciais de instituições renomadas e de agências espaciais, como a Nasa e a *Space*

Dynamics Laboratory - Utah State University e a Visiona Tecnologia Espacial. Essa ferramenta funciona como interface do usuário para controle e comando do sistema embarcado.

1.3 Objetivos

1.3.1 Objetivo geral

Este trabalho tem como objetivo principal desenvolver um sistema de visualização de dados de telemetria para o Centro de Controle de Missão do Projeto AlfaCrux e realizar simulações de envio de telecomando para o AlfaCrux. Outro objetivo é inserir o conhecimento sobre a ferramenta Cosmos dentro da Universidade de Brasília para que em missões futuras a parte de controle e comando possa ser desenvolvido por alunos da universidade.

1.3.2 Objetivos específicos

A fim de atingir o objetivo geral, o presente trabalho é dividido em etapas de desenvolvimento apresentadas abaixo:

- Simular o envio de dados genéricos para a ferramenta Cosmos;
- Prover um sistema de visualização de telemetria para a missão AlfaCrux capaz de receber mensagens de telemetria de uma estação de solo e realizar a visualização dos dados em tempo real;
- Criar janelas de telemetria com dados de engenharia e verificação de limites;
- Validar o sistema com dados reais de telemetria de missão AlfaCrux;
- Simular o envio de telecomandos do Cosmos para o AlfaCrux;

1.4 Organização do Trabalho

O presente trabalho é organizado de tal forma que conduza o leitor em uma sequencia lógica pelos conhecimentos teóricos necessários para o desenvolvimento desse estudo: fundamentos teóricos, metodologia desenvolvida, resultados e discussão e a conclusões da pesquisa.

Capítulo 2 - Fundamentos Teóricos - apresenta e contextualiza assuntos relacionados a satélites, sua carga útil e todo seu sistema, como o segmento solo e a cadeia de telemetria, telecomando e rastreio. Além disso, é apresentado informações da ferramenta Cosmos.

Capítulo 3 - AlfaCrux - descreve e contextualização o projeto AlfaCrux, especificando a missão, características técnicas e a contribuição do presente trabalho para esse *CubeSat*.

Capítulo 4 - Fundamentação Metodológica - descreve como foi realizado o trabalho em sua totalidade, para que possa ser replicado por terceiros. Na metodologia estão descritos os algoritmos utilizados para gerar as telas e para enviar um comando, bem como, os códigos desenvolvidos para o *software* de simulação de TM e simulação do AlfaCrux. Além disso, é descrito como o Cosmos interpreta as palavras chaves para a criação das telas.

Capítulo 5 - Resultados e Discussões - apresenta os resultados obtidos e a respectiva discussão. São apresentados os resultados da comunicação entre o Cosmos e o simulador de TM, o envio de telemetrias adquiridas da missão AlfaCrux para o Cosmos com o auxílio do simulador AlfaCrux, as telas desenvolvidas para visualização das telemetrias e os resultados do envio de comandos do Cosmos para o simulador.

Por fim, no capítulo 6 - Conclusões - são apresentadas as conclusões desse trabalho, avaliando se o objetivos foram alcançados e apresentando as possíveis melhorias, bem como as perspectivas para trabalhos futuros.

2 FUNDAMENTOS TEÓRICOS

2.1 Sistemas Espaciais

O primeiro foguete soviético com animais a bordo, lançado em 1951, e os voos subsequentes impulsionaram os estudos espaciais da antiga União Soviética. Em outubro de 1957, foi lançado em órbita terrestre o satélite Sputnik e quatro satélites *Zenit* não tripulados (CORPORATION, 2016). A partir disso, o número de satélites em órbita disparou, com mais de 6.000 satélites lançados nos anos seguintes. Esse avanço tecnológico foi impulsionado pela corrida espacial entre a União Soviética e os Estados Unidos (NASA, 2014a).

Em dezembro de 1958, os Estados Unidos lançam o primeiro satélite de comunicação, que forneceu um teste de um sistema de retransmissão de comunicação no espaço, trazendo de volta à Terra uma mensagem gravada. Em agosto de 1964, a NASA (do inglês, *National Aeronautics and Space Administration*) lançou o primeiro satélite geoestacionário, transmitindo ao vivo os Jogos Olímpicos de Tóquio em 1964. Durante esse período outros países também começaram a fazer lançamentos de satélites, como Inglaterra (1962), Canadá (1962), Itália (1964), França (1965) e Alemanha (1969) (NASA, 2014a).

Os avanços na área de eletrônica: circuitos integrados, computadores de alta capacidade, pequenos dispositivos com grande capacidade de memória, instrumentos miniaturizados e estruturas leves tornaram possível o desenvolvimento de satélites menores sem perder a da capacidade operacional (COUNCIL, 1994).

Um sistema espacial é composto por três segmentos de sistemas, coordenados de acordo com o objetivo da missão (LEY; WITTMANN; HALLMANN, 2009):

1. Segmento espacial: constituído pelos satélites (plataforma e sua carga útil) do sistema em órbita;
2. Segmento de transferência: fornece o transporte dos satélites construídos e testados em Terra para suas respectivas orbitas através de veículos lançadores;
3. Segmento solo: controla e monitora o Segmento Espacial, além de contribuir e processar os dados da carga útil;

Neste trabalho serão descritos em maior profundidade o Segmento Solo, em particular o Sistema de Controle de Missão SCM-αC para recepção e visualização de telemetria no Centro de Controle de Missão do Projeto AlfaCrux e para o envio de telecomandos.

2.2 Satélites

2.2.1 Conceitos

Satélite pode ser definido como um objeto que se move em torno de um objeto maior. Existem os satélites naturais - a Lua que se move ao redor da Terra - e os satélites artificiais - máquinas criadas pelo homem para alguma utilidade (NASA, 2014b; NASA, 2017).

Atualmente, a maioria dos países e territórios do mundo dependem das aplicações dos satélites artificiais para diversos serviços (PELTON; MADRY; CAMACHO-LARA, 2016a). Esses serviços podem ser divididos em seis categorias:

1. Comunicações por satélites (RICHHARIA, 1995);
2. Radiodifusão por satélites (MD, 2019);
3. Navegação, posicionamento e cronometria de precisão por satélite (BRUNO; ROMARIO, 2015);
4. Meteorologia de satélite geoestacionário e de órbita terrestre baixa (HURRICANES, 2020)
5. Satélites científicos;
6. Satélites de sensoriamento remoto ou observação da Terra;

Esses serviços citados acima são apenas o começo da indústria espacial, pois elas geram outras atividades relacionadas ao espaço, como o mercado multibilionário de antenas de estações terrestres, a indústria de veículos lançadores e mercados auxiliares como organizações de marketing e vendas. O mercado espacial tem crescido em proporções tão grandes que a indústria de telecomunicações por satélites teve uma receita anual em 2014 de aproximadamente US $ 203 bilhões (Fig. 2.1) (PELTON; MADRY; CAMACHO-LARA, 2016a).

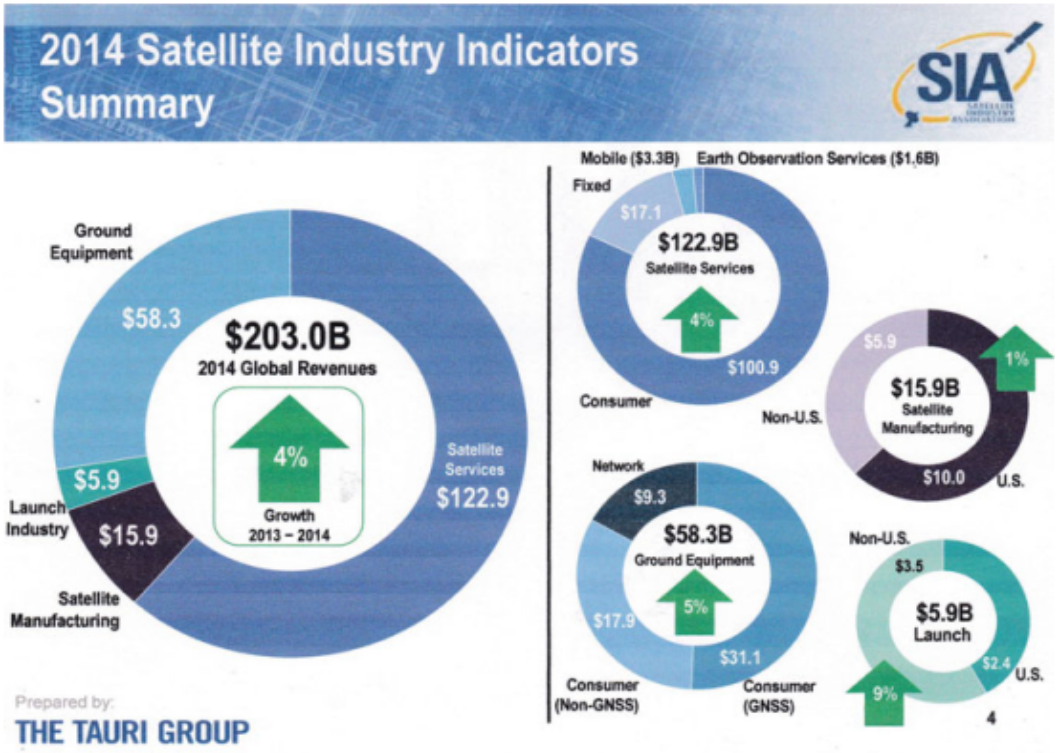


Figura 2.1 – Receita do mercado de comunicações por satélites em 2014
Fonte: (PELTON; MADRY; CAMACHO-LARA, 2016a)

Os satélites são divididos em suas partes: carga útil e plataforma. A carga útil é composta por equipamentos especializados que tem como objetivo cumprir a missão proposta para o voo. A plataforma precisa suportar o lançamento e auxiliar nas operações da carga útil (ROHLING, 2018).

Para o satélite funcionar adequadamente é necessário um conjunto de diferentes subsistemas com funções específicas (LEY; WITTMANN; HALLMANN, 2009). Cada satélite tem subsistemas diferenciados, na Fig. 2.2 é mostrado os subsistemas que geralmente compõem os satélites.

Cada subsistema tem uma função específica, no entanto, como o presente trabalho enfoca nas telemetria e nos telecomandos dos satélites, os quais serão detalhados apenas os subsistemas de Telemetria, Rastreio e Comando (TT&C, do inglês, *Telemetry, Tracking and Command*) e de Processamento de Dados (OBDH, do inglês, *On-Board Data Handling*). Além disso, será abordado sobre a Estação Terrestre e sobre o Centro de Controle de Missão.

Além da carga útil e da plataforma, se faz necessária a infraestrutura de solo para controle o monitoramento que seja capaz de realizar a comunicação com o satélite através do envio de telecomandos e da recepção, processamento e armazenamento da telemetria e de carga útil. As estações terrestres (GS, do inglês *Ground Stations*) são responsáveis por enviar os telecomandos e receber os dados da missão e os dados de telemetria (ROHLING, 2018). O centro de controle de missão (MCC, do inglês, *Mission Control Center*) pelo gerenciamento dos telecomandos

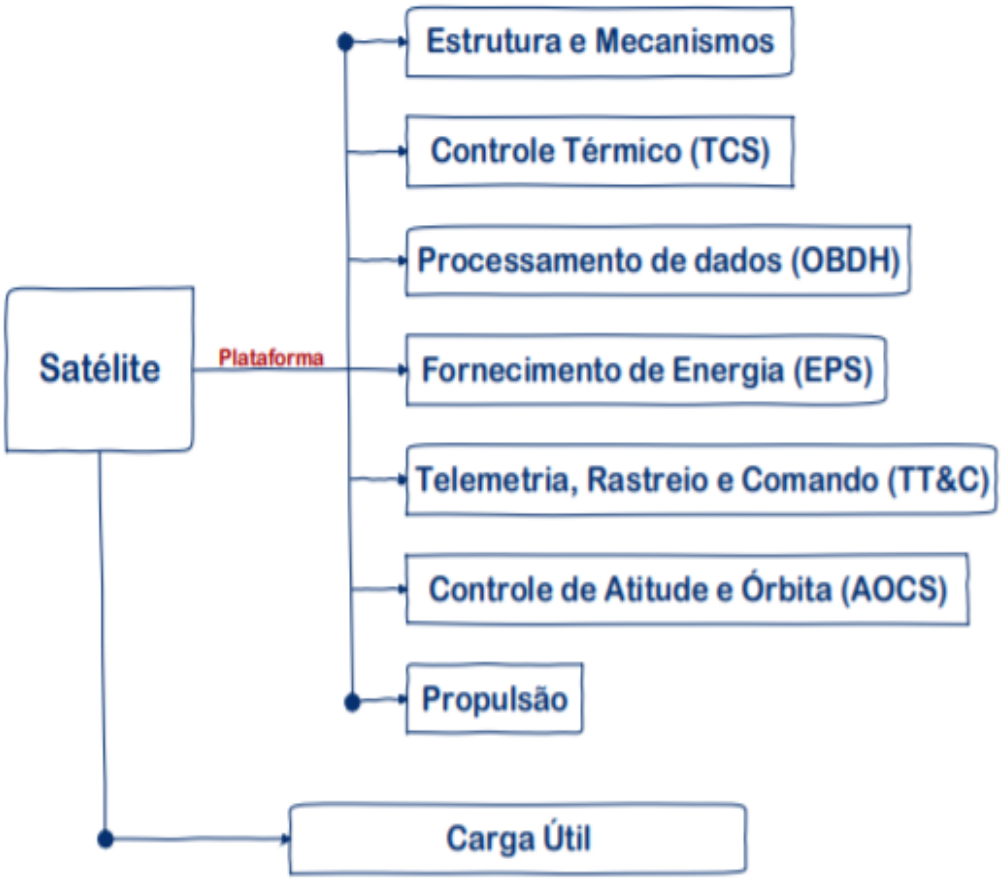


Figura 2.2 – Subsistemas dos satélites
Fonte: (RODRIGUES, 2016)

e pelo processamento e visualização desses dados (HARTWELL; LIGHTSEY, 2021). Dessa forma a GS e o MCC tem papel fundamental na comunicação com os satélites em órbita.

2.2.2 Classificação dos Satélites

Os satélites podem ser classificados por tipo de órbita, custo e massa, mas geralmente são agrupados de acordo com sua massa (NASA, 2020; CGEE, 2018; SHIROMA et al., 2011; SWEETING; UNDERWOOD, 2011; DUBOS; CASTET; SALEH, 2010). No entanto, nesta classificação podem ser encontrados valores diferentes, como apresentando nas Tabs. 2.1, 2.2 e 2.3. A Figura 2.3 mostra exemplos de cada classificação dos satélites de acordo com NASA (2020).

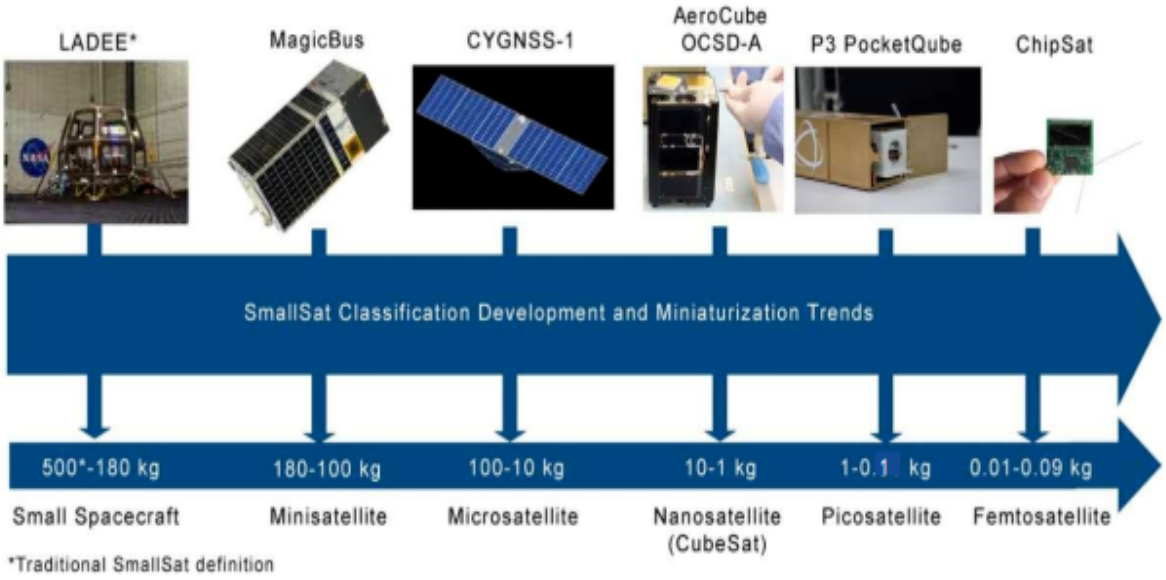


Figura 2.3 – Exemplos de cada classificação por massa de satélites de pequeno porte
Fonte: (NASA, 2020)

Tabela 2.1 – Classificação de satélites de pequeno porte por massa segunda a NASA

Classe	Faixa de massa [kg]
Minissatélite	100-180
Microssatélite	10-100
Nanossatélite	1-10
Picossatélite	0,1-1
Femtossatélite	0,01-0,09

Fonte: Adaptado de NASA (2020)

Tabela 2.2 – Classificação de satélites de pequeno porte por massa segundo livros e artigos

Classe	Faixa de massa [kg]
Minissatélite	100-500
Microssatélite	10-100
Nanossatélite	1-10
Picossatélite	0,1-1
Femtossatélite	<0,1

Fonte: Adaptado de Sweeting e Underwood (2011) e Wayne et al. (2011)

Tabela 2.3 – Classificação de satélites de pequeno porte por massa segundo o CGEE

Classe	Faixa de massa [kg]
Pequeno/médio satélite	100-1000
Microssatélite	10-100
Nanossatélite	1-10
Picossatélite	0,1-1

Fonte: Adaptado de CGEE (2018)

2.2.3 CubeSats

Dentro da classe de nanossatélites, foi criado uma ferramenta educacional, nomeada *CubeSat* (PESQUISA-FAPESP, 2019), o tipo de plataforma que impulsionou a maior participação internacional na última década (WAYNE et al., 2011).

O termo *CubeSat* é usado para designar um satélite de pequeno porte que tem o formato de um cubo (CGEE, 2018) com a dimensão de 10 cm x 10 cm x 10 cm (PESQUISA-FAPESP, 2019) com massa de até 1,33 kg (PROGRAM, Rev 12). Esses pequenos satélites tem a padronização da massa, forma e dimensão (JORDI; CLARK; WILLIAM, 2001) denominado 1U, com U representando uma "unidade"(PESQUISA-FAPESP, 2019). Esses cubos podem ser combinados formando unidades de 1,5U, 2U até 12U (NASA, 2020), como mostrado na Fig. 2.4.

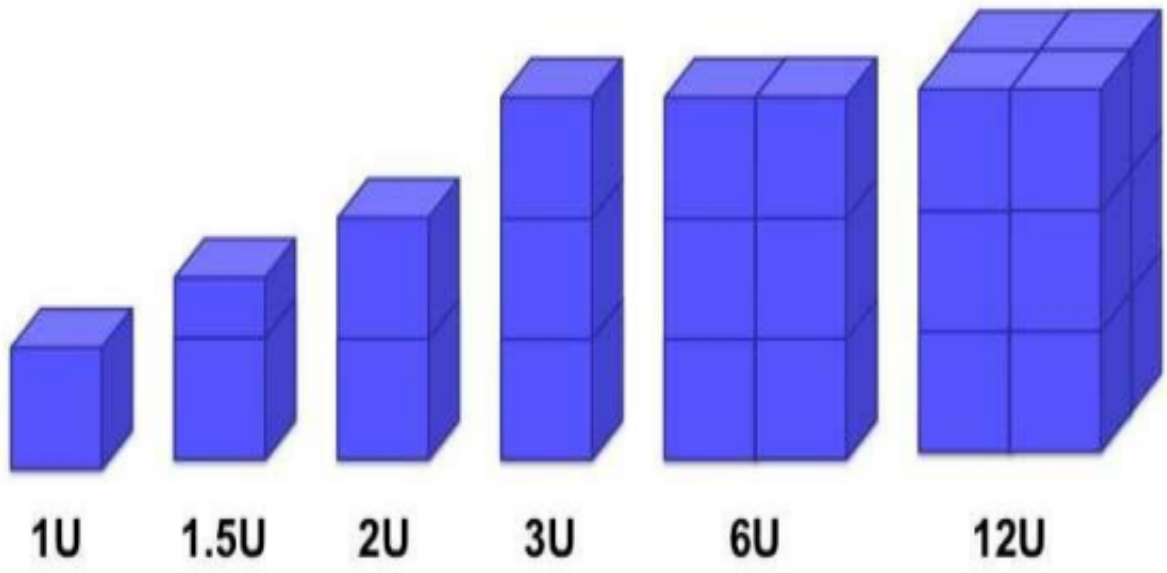


Figura 2.4 – Família dos *CubeSats* (1U-12U)
Fonte: (NASA, 2020)

Após o lançamento do satélite da antiga União Soviética Sputnik 1 e dos Estados Unidos Explorer 1, o lançamento de satélites no espaço tem tomado grandes proporções, atualmente com mais de 4.084 satélites em orbita (CONCERNED-SCIENTISTS, 2021). Além disso, o tamanho e o custo também cresceram consideravelmente (WERTZ; LARSON, 2005).

A criação de *CubeSats* foi motivada principalmente pelo preço elevado das missões espaciais, que engloba o desenvolvimento e construção do satélite, os sensores da missão, o veículo e as operações de lançamento e de missão, e testes e revisões extensivas (COUNCIL, 1994). Visto que o tamanho do satélite corresponde diretamente às despesas associadas com materiais, mão de obra e propelente para o veículo de lançamento, assim, a redução do tamanho reduz os custos (SHIROMA et al., 2011; CGEE, 2018).

Em concordância com as motivações citadas acima, o Prof. Jordi Puig-Suari da *California Polytechnic State University* (Cal Poly), San Luis Obispo e Prof. Bob Twiggs do *Space Systems Development Laboraroty da Stanford University*, iniciaram em 1999, o Projeto CubeSat. Atualmente, esse projeto é uma colaboração internacional de 100 universidades, escolas de ensino médio e empresas privativas (PROGRAM, 2020).

A Cal Poly desenvolveu o Poly Picosatellite Orbital Deployer (P-POD), um implantador de *CubeSat* padrão, suportando até três *CubeSats* padrão, com o objetivo de tentar garantir a segurança do *CubeSat* e proteger o veículo lançador, a carga útil primária e outros *CubeSats* (PROGRAM, 2020; JORDI; CLARK; WILLIAM, 2001).

2.3 Segmento Solo

O segmento solo é composto por toda a infraestrutura e os equipamentos de comunicação associadas com as estações fixas ou transportáveis. Esse segmento permite comandar e rastrear o satélite, receber e processar a telemetria, além de distribuir a informação para os controladores e para os usuários finais (RODRIGUES, 2016).

O segmento solo é composto por dois componentes principais: 1) centro de controle de missão (ou operações de missão (LEY; WITTMANN; HALLMANN, 2009)) - realiza as atividades de operações e prepara os dados de operações (procedimentos, documentação, descrição da missão) - e 2) estação terrena (ECSS, 2000) (ou rede de estação solo (LEY; WITTMANN; HALLMANN, 2009)) - constitui os principais elementos de infraestrutura de solo. A Figura 2.5 apresenta esses dois componentes: em azul, os elementos do MCC e em amarelo, os elementos da GS.

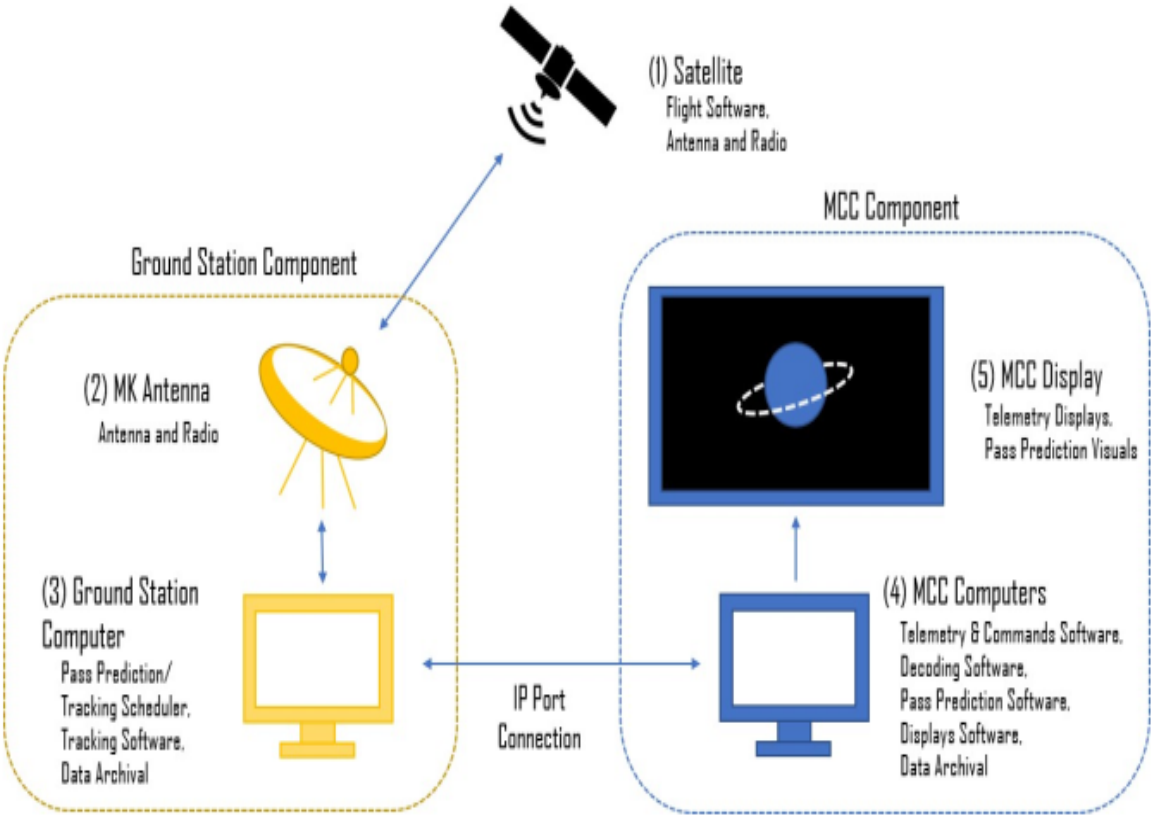


Figura 2.5 – Diagrama operacional do segmento solo para *CubeSats*
Fonte: (HARTWELL; LIGHTSEY, 2021)

2.3.1 Centro de Controle de Missão

Na maior parte, as operações são projetadas e conduzidas em um centro de controle e missão, no qual o *CubeSat* é monitorado e controlado e o tráfego de dados organizado. Além disso, contém interfaces dos dados para o fabricante e para seus usuários (LEY; WITTMANN; HALLMANN, 2009; ECSS, 2000).

As operações de voo são a parte central do MCC, as quais são realizadas em uma sala de controle, englobando o gerenciamento das tarefas reais de voo e manutenção do *CubeSat* em todas as fases da missão. As atividades variam com as diferentes fases e execução, a fase mais exigente em termos de pessoal e recurso é chamada fase de lançamento e operação inicial LEOP (do inglês, *Launch and Early Orbit Phase*) em relação as operações de rotina. Na fase LEOP, o *CubeSat* é ativado após a inserção no espaço sob condições extremas de temperatura, vácuo, radiação etc (LEY; WITTMANN; HALLMANN, 2009).

2.3.2 Rede de Estação Solo

A estação solo é a interface direta com o segmento espacial com o MCC. Além de fazer a comunicação com o *CubeSat* e fazer simulações para determinar sua posição, a estação solo pode ter as seguintes funções (FORTESCUE; STARK; SWONERD, 2003):

- Operações de telemetria: adquire e registra os dados de telemetria transmitidos pelo *CubeSat*;
- Operações de comando: transmitem os telecomandos que controlam as várias funções do *CubeSat*;
- Operações de rastreio: rastreia o satélite durante a passagem sobre a estação podendo realizar medidas de distância e medidas de Range Rate (Doppler);
- Operações de Monitoramento e Controle dos equipamentos da estação.

Segundo ECSS (2000), as estações solos são classificadas em relação à sua utilização da seguinte forma:

- GSTS-SSC (*Ground Station System - Space Segment Control*): apoia o controle do segmento espacial (fornece telemetria, rastreio e telecomando) para o *CubeSat*;
- GSTS-ME (*Ground Station System - Mission Explotation*): apoia a exploração da missão.

O elemento mais visível do caminho de transmissão de e para o *CubeSat* são as antenas. A comunicação entre a GS e o satélite pode utilizar várias bandas de frequência, como visto na Seção 2.4, cuja atribuição está sujeita à coordenação internacional pela ITU (do inglês, *International Telecommunication Union*) (LEY; WITTMANN; HALLMANN, 2009).

O perfil da missão de uma estação terrestre também depende da fase da missão. Durante a preparação da missão, a compatibilidade com os componentes de comunicação do *CubeSat* e a configuração das interfaces de dados têm alta prioridade. Já no início da fase de LEOP, no qual acontece a separação com o veículo lançador, as atividades mais importantes são a detecção da

posição do *CubeSat* e os contatos frequentes e duradouros com o satélite (LEY; WITTMANN; HALLMANN, 2009).

A Figura 2.6 mostra redes globais de estações terrestres estabelecidas, para assegurar um contato mais frequente ou interrupto da GS com o *CubeSat*, especialmente na fase LEOP crítico, pois a janela de visibilidade dos satélites em órbita terrestre baixa (LEO, do inglês, *Low Earth Orbit*) para uma estação solo normalmente duram alguns minutos, depois disso, o satélite pode ficar sem contato com o solo por horas (LEY; WITTMANN; HALLMANN, 2009).

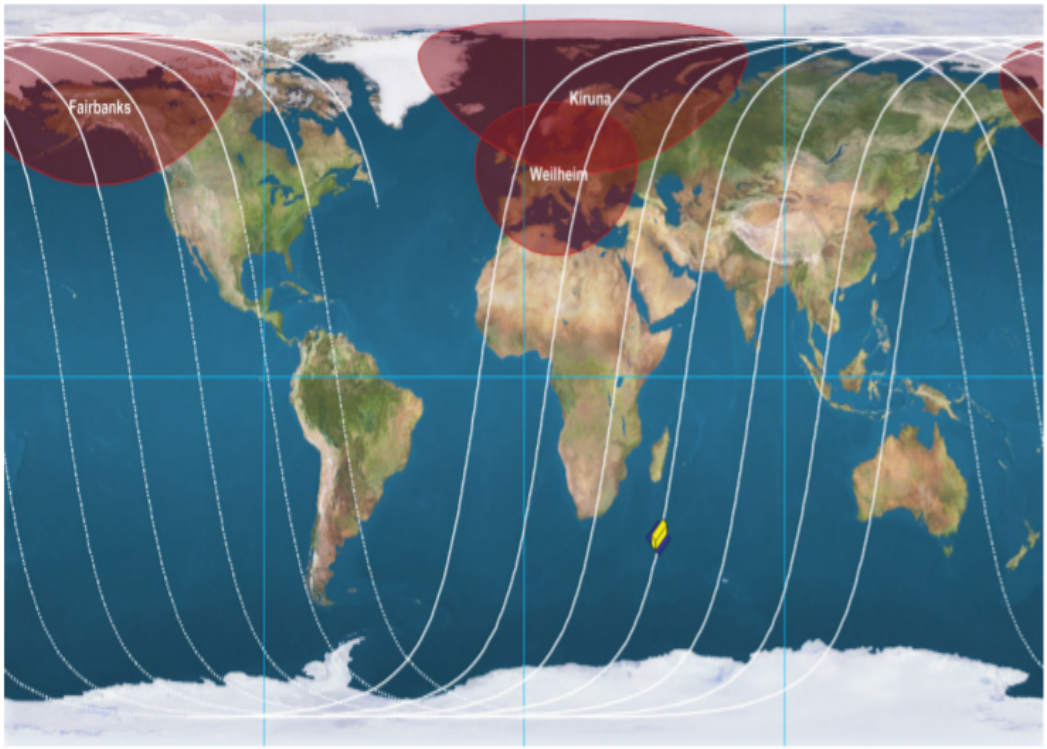


Figura 2.6 – Rastreamento global do satélite DLR em órbita BIRD durante o LEOP
Fonte: (LEY; WITTMANN; HALLMANN, 2009)

2.4 Cadeias de Telemetria, Telecomando e Rastreio

Segmento Espacial e Segmento Solo interagem através de cadeias de Telemetria, Telecomando e Rastreio representando a interface solo bordo, como apresentando na Fig. 2.7. Também representada pela cadeia de comunicação solo bordo incluindo os protocolos de comunicação. Também representada pela arquitetura de comunicações usada no sistema.

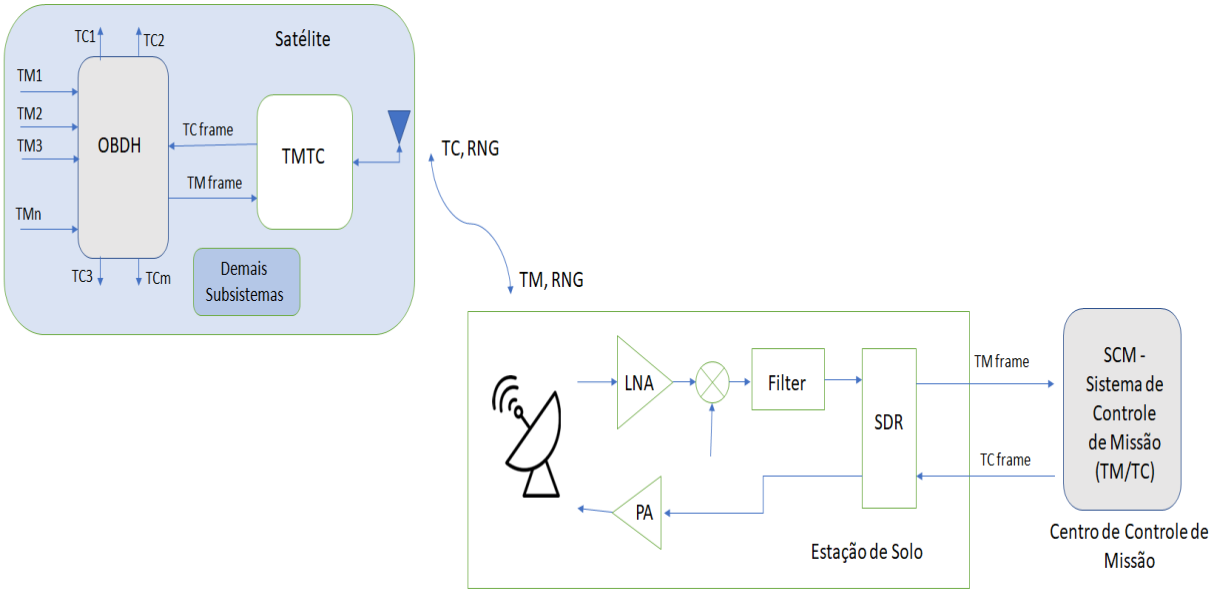


Figura 2.7 – Interação entre o segmento espacial e o segmento solo por meio d o TT&C
Fonte: Autor

A comunicação entre Segmento Espacial e Segmento Solo ocorre por meio da ligação espacial utilizando um protocolo de comunicação espacial. Os protocolos mais usuais em *CubeSats* são o AX.25 e o CCSDS (do inglês, *Consultative Committee for Space Data Systems*).

O AX.25 é um protocolo de pacote de radioamador que está em conformidade com a ISO (do inglês, *International Standards Organization*) 3309, 4335 e 7809 HDLC (do inglês, *High-level Data Link Control*) e segue a terminologia desses documentos (BEECH; NIELSEN; TAYLOR, 1998).

No protocolo AX.25, os primeiros *bits* enviados são os LSB (do inglês, *Least Significant Bit*) e os primeiros *Bytes* de *frames* enviados são os mais significativos. Os satélites usam o quadro de *frame* não numerado (UI-Frame, do inglês *Unnumbered Information Frame*), essa estrutura é apresentada na Fig. 2.8 (LEFFKE, 2018).

FLAG	AX.25 Transfer Frame Header (128 bits)				Information Field	Frame Check Sequence	FLAG
	DESTINATION ADDRESS	SOURCE ADDRESS	Control Bits	Protocol Identifier			
8	56	56	8	8	0-2048	0-2048	8

Figura 2.8 – Estrutura do envio de *Frames* do protocolo AX.25
Fonte: (LEFFKE, 2018)

Os *FLAGs* são usados para marcar o início e o fim da telemetria, ajudam o receptor a sincronizar e tem o valor fixo de 01111110 (0x7E) (Fig. 2.9). O *DESTINATION ADDRESS* e o *SOURCE ADDRESS* são compostos por 6 *Bytes* indicativos e 1 *Byte* SSID (do inglês, *Service Set IDentifier*). Os 6 *Bytes* indicativos são divididos em 7 *bits* ASCII e um *bit* que indica se há mais dados ou não (0 = há mais dados, 1 = fim dos dados). Os *bits* 5, 6 e 7 do SSID são fixos (011) e os *bits* das posições de 1 a 4 são valores inteiro de 16 bit, normalmente, 0000, veja na Fig. 2.10 (LEFFKE, 2018).

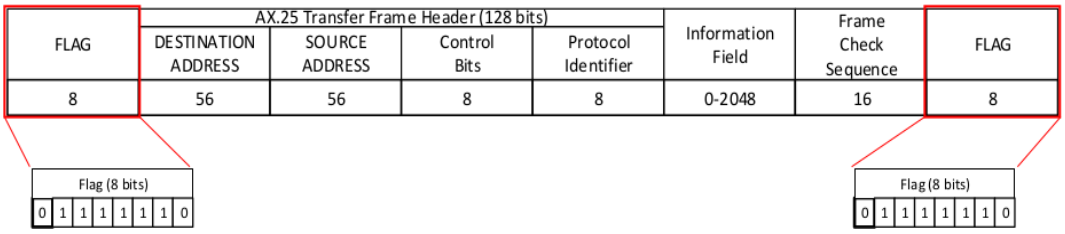


Figura 2.9 – Estrutura dos *bits* das *FLAGs*
Fonte: (LEFFKE, 2018)

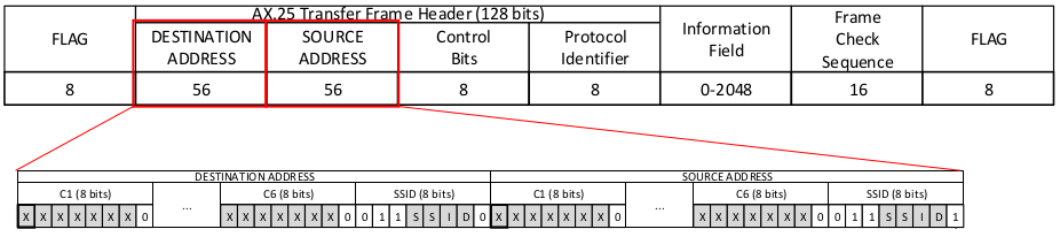


Figura 2.10 – Estrutura dos *bits* do *DESTINATION ADDRESS* e do *SOURCE ADDRESS*
Fonte: (LEFFKE, 2018)

O *Control Bits* indica qual tipo de *frame* está sendo enviado, para o *UI-Frame* deve ser fixado em 00000011 (0x03). E o *Protocol Identifier* indica qual tipo da camada 3, para nenhuma camada 3 implementada deve ser fixado em 11110000 (0xF0), mostrado na Fig. 2.11. O *Information Field* contém os dados de telemetria, de telecomando e de missão, seu tamanho máximo é de 256 *Bytes*. O *Frame Check Sequence* é usado para detectar erros de *bits* baseado no CRC (do inglês *Cyclic Redundancy Check*) (LEFFKE, 2018).

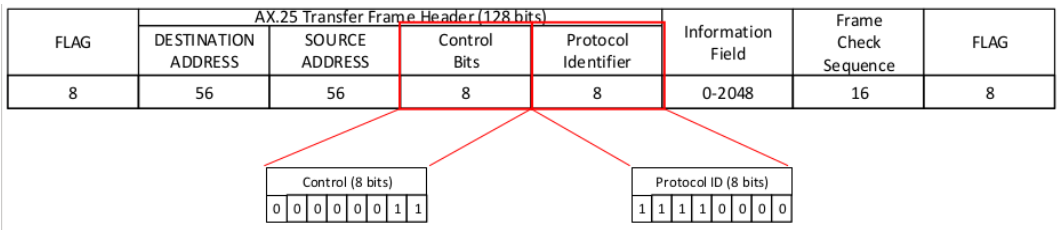


Figura 2.11 – Estrutura do *bits* do *Control Bits*
Fonte: (LEFFKE, 2018)

A maioria dos protocolos de comunicação espacial do CCSDS é definido com base no Modelo de Referência Básico OSI (do inglês, *Open Systems Interconnection*). No relatório

CCSDS (2014), o CCSDS utiliza esse modelo para categorizar os protocolos de comunicação espacial para cinco camadas do modelo ISO: camada física, camada de ligação de dados, camada de rede, camada de transporte e camada de aplicação. Os protocolos das camadas de sessão e de apresentação dificilmente são utilizadas em ligações espaciais.

A camada física utiliza o Sistema de Rádio Frequência e Modulação, ou seja, define a frequência de transmissão, o tipo de modulação e a taxa de bits a ser transmitida entre a naves espaciais e as estações terrestres CCSDS (2014).

A camada de ligação de dados é dividida em duas subcamadas: subcamada de protocolo de ligação de dados e subcamada de sincronização e codificação do canal. A subcamada de protocolo de ligação de dados define o método de transferência de unidade de dados conhecidas como *Transfer Frames*. A segunda subcamada define o método de sincronização e codificação de canais para a transferência das unidades de dados (CCSDS, 2014).

Os protocolos que definem a transferência ou o encaminhamento de pacotes de dados entre as subredes a bordo ou entre as subredes da estação terrestre fazem parte da camada de rede. O CCSDS fornece dois protocolos para essa camada: *Space Packet Protocol* e *Encapsulation Service*. No entanto, existe um protocolo, não pertencente ao CCSDS, específico para *CubeSats* para essa camada, o CSP (do inglês, *CubeSat Space Protocol*) (CCSDS, 2014).

Os protocolos da camada de transporte são responsáveis por transportar e regular o fluxo de dados entre o segmento espacial e o segmento solo de forma confiável, fornecendo o serviço de transporte de ponta a ponta ou fim-a-fim. A última camada, a camada de aplicação, fornece serviços de aplicação diretamente para o usuário, ou seja, são normas que abordam a utilização dos pacotes de telemetria e de telecomando (CCSDS, 2014).

O presente trabalho se aplica no Sistema de Controle de Missão apresentado na Fig. 2.7, o sistema proposto recebe os *frames* de telemetria e apresenta os dados em telas personalizadas de acordo com seus limites e unidades de engenharia, bem como envia *frames* de telecomandos para a estação solo para que ela transmita para o *CubeSat*.

2.4.1 Cadeia de Telemetria

Outro subsistema vital para todos os satélites é o TMTC (do inglês, *Telemetry and Telecommand*) mostrado na Figura 2.12, independente da aplicação, pois a sua função é fazer a comunicação entre o satélite e as instalações no solo. Sem esse subsistema, o satélite fica incomunicável e, essencialmente, a missão é finalizada (PELTON; MADRY; CAMACHO-LARA, 2016b). É um sistema responsável pela transmissão dos dados da carga útil e de manutenção do satélite, bem como recepção a bordo de telecomando, garantindo, assim, que o satélite possa ser monitorado e controlado corretamente a partir da infraestrutura de solo (GUEST, 2016; DELMONDES, 2019).

A coleta de medições e leitura dos instrumentos embarcados ligados ao OBDH, que por

sua vez está ligado ao TMTC, são necessários para monitorar a saúde e o *status* de todos os subsistemas do *CubeSat* (GUEST, 2016). É necessário a coleta, o processamento e a transmissão desses dados do satélite para a estação solo e posteriormente para o centro de controle e comando, como apresentando na Fig. 2.12 (GUEST, 2016; BARY; RAYMOND; KENNETH, 1993). Os principais dados relacionados a saúde e o *status* do *CubeSat* são: O *status* dos recursos (ex: saúde das baterias), a atitude do satélite, o modo de operação para cada subsistema (ex: se a roda de reação está ligada ou desligada) e a integridade de cada subsistema (ex: saída dos painéis solares).

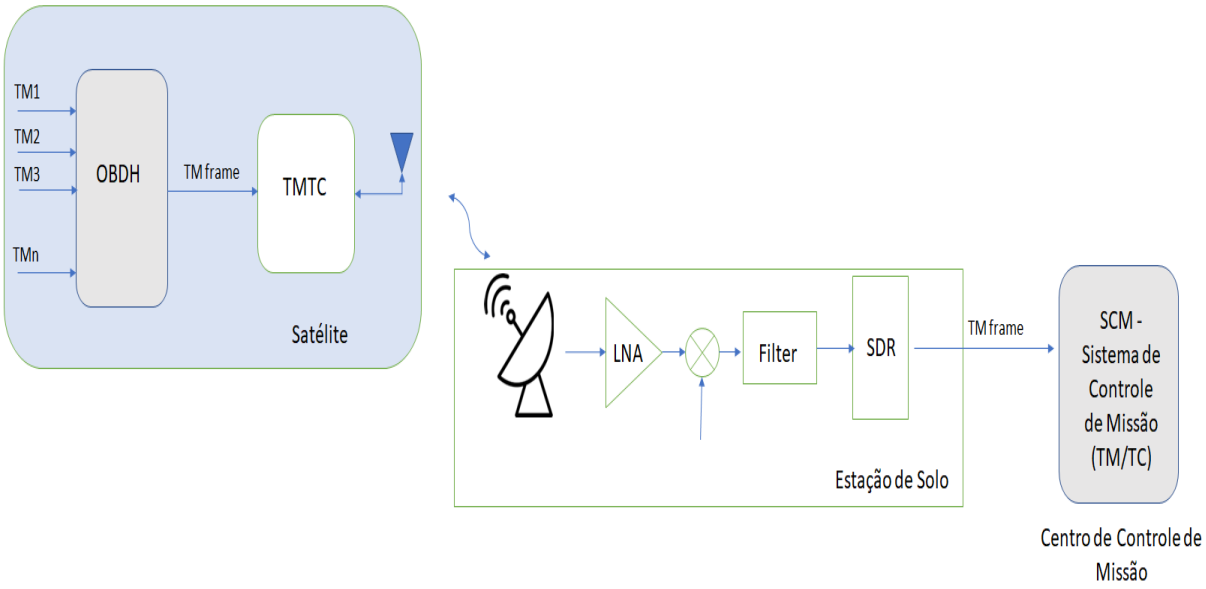


Figura 2.12 – Diagrama do envio dos dados do OBDH até o MCC
Fonte: Autor

Todos os dados são coletados com vários sensores, como termistores, acelerômetros, giroscópios e outros transdutores, fornecendo as respostas em medidas de resistência, capacitância, corrente ou tensão. Posteriormente, é necessário o processamento desses dados para converter as medidas analógicas em digitais e formatar as medidas em um padrão para a transmissão (GUEST, 2016).

Visto que é extremamente caro ter uma estação terrestre que tenha conectividade constante com a equipe de controle em solo, pode ser necessário o armazenamento dos dados de telemetria (GUEST, 2016). Assim, o sistema embarcado deve ser capaz de processar e armazenar os dados dos sensores a bordo até a abertura de uma janela viável de comunicação, devido a dificuldade de acesso contínuo com a equipe de controle. Os dados armazenados são transmitidos com informações pertinentes, como quando cada dado foi coletado. Esses dados são essenciais para a equipe de controle em solo, principalmente em caso de anomalias (GUEST, 2016).

A NASA criou uma alternativa para o armazenamento de dados, *Tracking and Data Relay Satellite System* (TDRSS), que consiste em três estações terrestres localizadas no Complexo White Sands (WSC) no sul do Novo México, o Terminal Terrestre Remoto Guam (do inglês, GRGT) na Estação Naval de Telecomunicações e Computadores de Guam e o Centro de Controle de Rede localizado no Space Flight Center in Greenbelt, Maryland, os quais podem retransmitir a telemetria de qualquer satélite em órbita terrestre baixa para locais específicos da Terra. Esse sistema também pode ser usado para fornecer *uplinks* necessários para a emissão de telecomandos a um satélite (NASA, 2021; WILLIAM, 2003).

As frequências geralmente utilizadas para o sistema de telemetria dos Serviços de Radioamador estão descritas na Tab 2.4. No entanto, as bandas de frequências não são limitadas a essas, outras bandas também podem ser empregadas para diferentes tipos de serviços de comunicações (Operação Espacial (SOS), Serviços Fixos por Satélite, Serviço de Exploração da Terra por Satélite, e outros conforme os Regulamentos de Radio da ITU e o Plano de Atribuição, Destinação e Distribuição de Frequências no Brasil (PDFF) da Anatel. Os sistemas de telemetria que utilizam frequências mais elevadas (banda Ku, Ka) devem levar em consideração a atenuação da chuva no projeto. Em relação ao erro de bit, as comunicações tendem a ter um erro de aproximadamente 10^{-5} (GUEST, 2016; WERTZ; LARSON, 1999).

Tabela 2.4 – Bandas para a trasmissão de telemetrias

	Frequência
VHF	29,52 - 29,70 MHz
UHF	222,25 - 225 MHz
Banda S	1,27 - 1,28 GHz
Banda C	5,65 - 5,85 GHz
Banda Ku	11,7 - 12,2 GHz
Banda Ka	26 GHz

Fonte: Adaptado de LABRE-RS (2021)

2.4.2 Cadeia de Telecomando

Outra cadeia funcional importante é a de Telecomando. O telecomando é gerado pelo Sistema de Controle de Missão de acordo com o protocolo estabelecido, registrado e encaminhado ao equipamento de Banda Base ou SDR da estação de solo para que seja modulado e transmitido para o satélite. No satélite o subsistema TT&C deve receber os telecomandos transmitidos pela estação solo e encaminhar para o OBDH para a sua interpretação e execução a bordo. Dependendo do projeto do OBDH, pode ser implementado automatismo a bordo que implementa comandos predefinidos mediante o reconhecimento de condições específicas (GUEST, 2016).

A automação permite que ações instantâneas sejam realizadas a bordo, essa autonomia varia de acordo com a aplicação do satélite e com a sofisticação dos *softwares*. Segundo Pisacane (2005), a automação tem três requisitos específicos:

1. Capacidade do *software* embarcado identificar adequadamente a telemetria e o comando que indica que um subsistema está agindo de forma incorreta e a capacidade de identificar e processar a resposta correta;

2. Capacidade da telemetria e do comando se comunicarem entre si e passarem informações de um lado para o outro quase que instantaneamente;

3. Capacidade de diagnóstico que permita ao sistema de telemetria determinar se uma leitura anormal é causada por outro subsistema ou por erros próprios do TT&C.

Nesse contexto, tem-se que os comandos são extremamente importantes, pois podem reconfigurar tanto o satélite, quanto seus subsistemas, a fim de responderem a cada circunstância da missão. Eles podem conter a comutação de subsistemas e de componentes ligados ou desligados, podem também alterar condições de operações de componentes. Por exemplo, os telecomandos podem controlar a orientação e a atitude do satélite, podem também controlar estruturas, como painéis solares. Por fim, os telecomandos podem ser usados no computador de bordo para o controlar os componentes de forma contínua (GUEST, 2016).

Os enlaces para o sistema de telecomando são calculados de forma similar ao de telemetrias. A taxa de erro de bit tem uma magnitude menor comparada com as comunicações de telemetria, de aproximadamente 10^{-6} (KEEESEE, 2003), devido a importância de garantir que os telecomandos emitidos pelo solo sejam reconhecidos corretamente pelo subsistema TT&C.

O satélite recebe e demodula o sinal de telecomando, utilizando o decodificador de telecomando, a lógica de telecomando - valida o telecomando e o rejeita se houver alguma incerteza quanto à sua autenticidade - e o circuito de interface - que implementa o telecomando validado e o conecta a outro sistema no satélite. Caso o sistema seja mais sofisticado em seus processos de segurança, ele pode demandar de alguns telecomandos de solo para serem autenticados (GUEST, 2016).

Nesse sentido, para elucidar os segmentos complementares, primeiramente, o decodificador de telecomando tem a função de reproduzir mensagens e produzir sinais de bloqueio, para isso, ele recebe e processa todos os telecomandos de entrada de fontes. O decodificador conta com um esquema, o qual estabelece um fila de prioridade para os telecomandos. Normalmente, as mensagens de comando contém bits de verificação de entrada, de sincronização, e de detecção de erro (GUEST, 2016).

O telecomando, propriamente dito, traz o tipo de telecomando e o endereço no satélite, e, em sua maioria, tanto mensagens, como o próprio telecomando devem ser analisados. Dentre os telecomandos, existem alguns tipos como para alterar o nível de saída de um componente ou para solicitar dados para um componente, podendo envolver uma variedade de sequências de eventos (GUEST, 2016).

Já a lógica de telecomando, que tem a função de verificar e validar o telecomando decodificado, a fim de garantir o envio de telecomandos corretos ou válidos. Nessa etapa, o momento do telecomando tem de ser validado e o telecomando em si, autenticado, para esse processamento do comando. o subsistema TT&C e o OBDH ativam os circuitos, conforme as necessidades do tipo de telecomando a ser processado. Caso haja dúvidas sobre sua autenticidade, o comando pode ser rejeitado e, no caso de problemas, pode ser preciso mudar as restrições no *software* com vistas a permitir a execução de telecomandos com risco maior (GUEST, 2016).

2.4.3 Cadeia de *Ranging* e *Range-Rate*

A terceira função do subsistema TT&C é responsável por permitir a determinação de distância (*ranging*) pela estação solo e com essas medidas realizar a determinação de órbita. Outra possibilidade é permitir a medição de velocidade radial (*range rate*) pela estação de solo com transmissão de sinal sincronizado com a portadora de subida. Com esses dados de medida de distância e ou medidas de velocidade do satélite, o Centro de Controle pode determinar a orbita e depois realizar a propagação de órbita.

A determinação de distância (*ranging*) é normalmente obtida por meio do uso de tons ou

sequência pseudo aleatória (Pseudo Noise). O tom ou a sequência pseudo aleatória é modulado para a frequência de *uplink* e o subsistema TT&C retransmite o mesmo tom ou sequência pseudo aleatória para o *downlink*. O cálculo da distância entre a estação terrena e o *CubeSat* é feito utilizando o tempo de ida e volta do tom ou da sequência. Com as medidas de distância, a localização real do *CubeSat* pode ser determinada usando *software* de dinâmica de voo, através da determinação de órbita e sua propagação, gerando informação de previsão de passagem sobre a estação de solo e fornecendo o apontamento do satélite, azimute e em elevação para a estação solo (GUEST, 2016).

Outro método utilizado para determinar a órbita de um satélite é através da medida de velocidade radial (*range-rate*) ou o deslocamento Doppler. Quando o satélite está se aproximando de uma GS, a frequência recebida é maior que a frequência transmitida. Quando o satélite se afasta da GS, a frequência recebida é menor que a frequência transmitida. Um exemplo da utilização desse fenômeno é o Sistema Argos, mostrado na Fig. 2.13, cujo *payload* DCS a bordo mede o deslocamento dos sinais transmitidos por plataformas de coleta de dados (*Argos Transmitter*) em solo, em adição às mensagens enviadas da plataforma. Com os dados de Doppler, o Argos pode determinar a locação do transmissor em Terra. Exemplos de algoritmos utilizados para determinar a localização de transmissores são análise de mínimos quadrados e filtro de Kalman (GUEST, 2016).

Como informação complementar, a determinação de órbita pode também ser efetuada através de medidas de apontamento da antena (tempo, azimute e elevação) durante uma passagem de um satélite com uso de sistema de rastreio *autotracking* da estação de solo. Outra forma de realização a determinação de orbita seria uso de um sensor GPS a bordo.

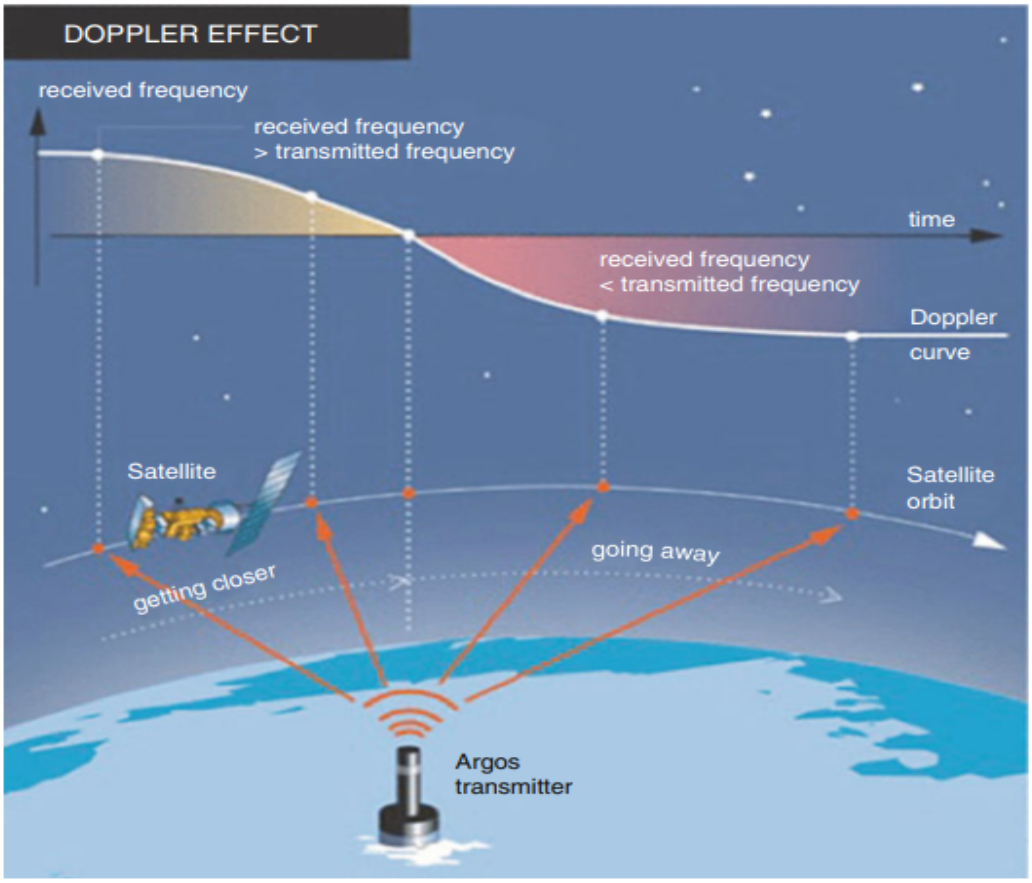


Figura 2.13 – Satélite Argos utilizando o efeito Doppler para localizar um transmissor Argos no solo

Fonte: (ARGOS, 2016)

2.5 Sistema de Controle do Centro de Controle de Missão

O Sistema de Controle de Missão é um sistema usado para o registro, processamento e visualização de dados. Fisicamente, é acomodado em uma sala de controle com projeções em vídeo *wall* e monitores de vídeos para exibir informações de interesse geral e em tempo real (LEY; WITTMANN; HALLMANN, 2009).

Este é um componente do MCC que deve reconhecer as telemetrias vindas da estação solo e organizar os telecomandos para enviar para a estação solo, o *Telemetry and Command Software*. Esse *software* pode passar as informações de telemetrias para outro aplicativo de visualização que possibilitam além de visualização de telemetria, analisar o desempenho de um subsistema ao longo da órbita (HARTWELL; LIGHTSEY, 2021). A ferramenta Cosmos contém vários aplicativos e dentre eles, estão o *Telemetry and Command Server*, o *Telemetry Viewer* e o *Command Sender* que possibilitam realizar essas duas atividades.

2.5.1 Cosmos

A ferramenta Cosmos, da Ball Aerospace, é um sistema de comando e controle que inclui o recurso de visualização de dados para sistemas embarcados podendo ser usado durante as fases de Montagem, integração e testes do satélite e durante a operação em órbita. Essa ferramente é composta por 18 aplicativos, descritos no Quadro 2.1, e durante a inicialização do Cosmos são apresentados em uma janela nomeada *Launcher* (Fig. 2.14) (BALLAEROSPACE, 2021e).

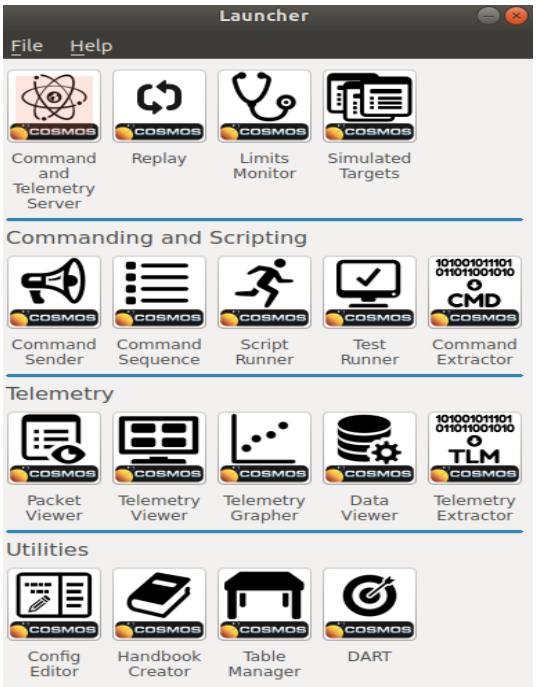


Figura 2.14 – Janela inicial do Cosmos, Launcher
Fonte: Adaptado de BallAerospace (2021e)

Esses 18 aplicativos que compõem o Cosmos se relacionam entre si e são agrupados em quatro categorias amplas: comando e *scripts* em tempo real, visualização de telemetria em tempo real, análise *offline* e serviços de utilidade pública, como mostrado na Fig. 2.15.

Essa ferramenta é um *software* livre que pode ser instalado seguindo os passos descritos no guia de instalação, encontrado em BallAerospace (2021c). A instalação dá o acesso ao ambiente DEMO. No GitHub BallAerospace (2021b) é possível visualizar o código fonte, dar sugestões e fazer perguntas. Além disso, cada um dos aplicativos apresentados no Quadro 2.1 é construído usando bibliotecas Cosmos que estão disponíveis para uso como uma estrutura para desenvolver aplicativos de projeto personalizados. O entendimento do seu funcionamento depende da compreensão de alguns termos importantes, apresentado no Quadro 2.2.

Os arquivos dentro do diretório da ferramenta têm locais bem definidos, a estrutura de diretório é mostrado na Fig. 2.16. Os arquivos de configuração utilizados para configurar o Cosmos são feitos através de textos simples, ou seja, salvos na extensão .txt (BALLAEROSPACE, 2021a).

Quadro 2.1 – Descrição dos 18 aplicativos do Cosmos

Command and Telemetry Server	Fornece *status* sobre as interfaces e sobre os *targets* (alvos), além de fornecer atalhos rápidos para a visualizar comandos brutos e formatados e pacotes de telemetria
Replay	Simula um servidor de comando e telemetria para reprodução de arquivo de *log* de pacote de telemetria
Limits Monitor	Monitora as telemetrias com limites definidos e mostra os itens que que violam os limites
Simulated Targets	Abre o terminal para simular um alvo
Command Sender	Interface gráfica para enviar manualmente comandos individuais
Command Sequence	Envia uma sequência de comando por meio de uma classe Ruby
Script Runner	Executa *scripts* de comandos, fornecendo destaque da linha em execução
Test Runner	Fornece uma estrutura de alto nível para teste de nível de sistema, incluindo geração de relatório de teste
Command Extractor	Extrai itens de comando em arquivos separados por vírgula ou tabulação, além de possuir opções para controlar a saída para pós-processamento no Excel ou Matlab
Packet Viewer	Possibilita a visualização de pacotes individuais de telemetria em tempo real que não requer configuração para exibir os valores atuais
Telemetry Viewer	Possibilita a visualização de telas personalizadas de telemetria
Telemetry Grapher	Permite a representação gráfica de dados de telemetria
Data Viewer	Permite a visualização de dados de pacote no passado e em tempo real
Telemetry Extractor	Extrai itens de telemetria em arquivos separados por vírgula ou tabulação, além de possuir opções para controlar a saída para pós-processamento no Excel ou Matlab
Config Editor	Permite o usuário editar arquivos de configuração COSMOS por meio de textos
Handbook Creator	Cria um editor de arquivo em html ou pdf dos pacotes de telemetrias configurados
Table Manager	Um editor de arquivo binário que pode ser usado para construir tabelas de configuração ou outros dados binários
DART	Suporta o armazenamento de longo prazo

Adaptado de BallAerospace (2021e)

Não é necessário alterar todos os diretórios para receber as telemetrias, enviar os comandos e visualizar as telemetrias. Posto isso, será detalhado no Quadro 2.3 apenas os principais diretórios que precisam ser configurados.

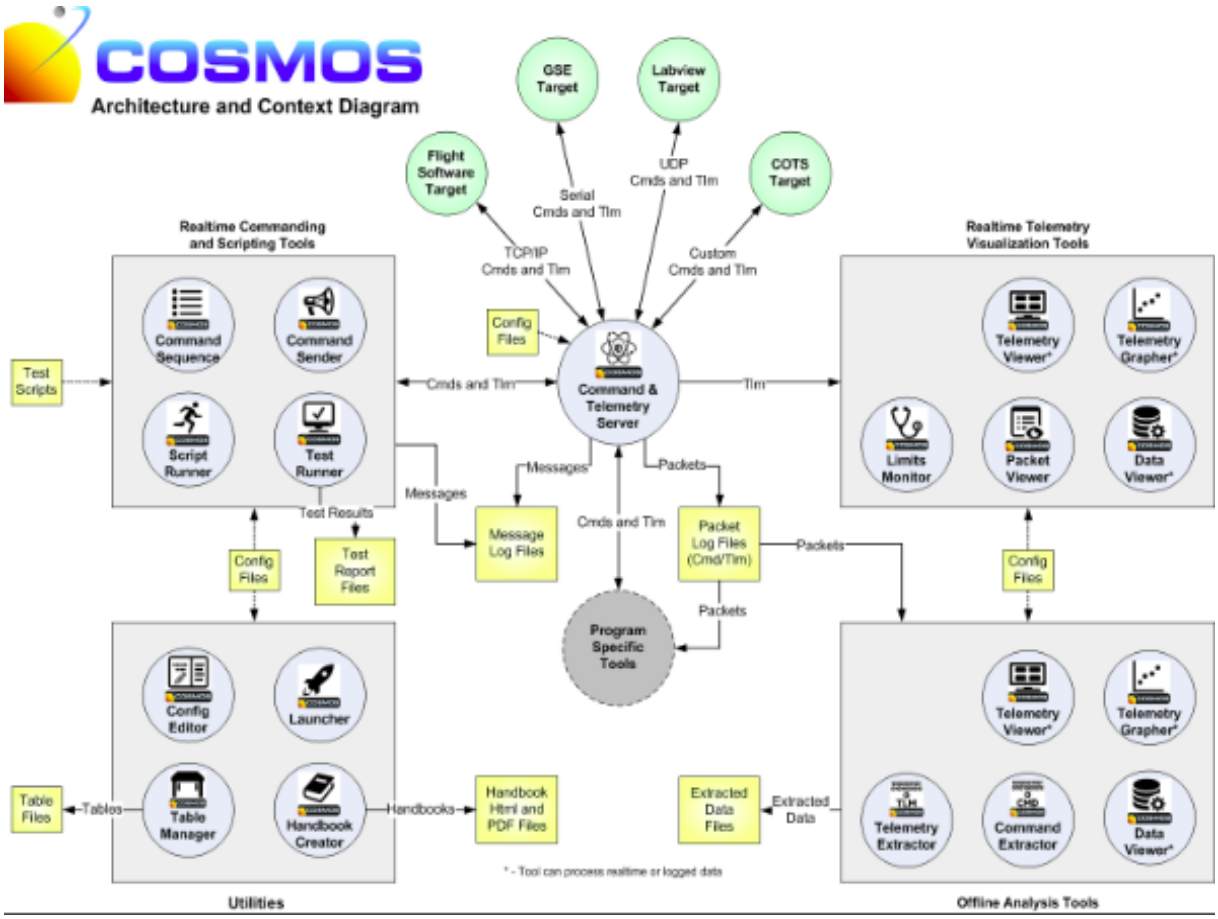


Figura 2.15 – Arquitetura geral do Cosmos
Fonte: (BALLAEROSPACE, 2021e)

2.5.2 Telemetria

A visualização de telemetrias dentro do Cosmos pode ser feita em forma telas, gráficos ou em uma lista de telemetrias de acordo com os pacotes. O presente trabalho tem o enfoque em apresentar os dados recebidos de um *CubeSat* por meio de telas com o auxílio do aplicativo *Telemetry Viewer*. As telas permitem a visualização dos dados de forma mais dinâmica e também podem ser inseridos gráficos.

No *Telemetry Viewer* é possível personalizar telas de acordo com que o fabricante ou usuário do *CubeSat* deseja visualizar. Há dois tipos diferentes de arquivos usados para configurar o aplicativo *Telemetry Viewer*: os arquivos de definição de tela e um arquivo de configuração que permite à ferramenta saber quais telas estão disponíveis e como estão organizadas (BALLAEROSPACE, 2021g).

Os arquivos de definição de tela de telemetria são usados para definir o conteúdo das telas, por exemplo, quais dados apresentar, como esses dados vão ser apresentados, em qual posição da tela vai ser mostrado cada dado, bem como, configurações da fonte, número de abas, tamanho da tela, entre outros. Esses arquivos assumem a forma geral de uma palavra chave *SCREEN* seguida por uma série de palavras chaves. Os arquivos de definição de tela de um determinado

Quadro 2.2 – Terminologias importantes do Cosmos

Target	O alvo é um sistema embarcado que o *Command and Telemetry Server* (CTS) se conecta para enviar comandos e/ou receber telemetrias
Command	Um pacote de informações dizendo a um alvo para executar uma ação de algum tipo
Telemetry Packet	Um pacote de informações que fornece o *status* de um *target*
Ruby	Linguagem de programação dinâmica usada para escrever os aplicativos e bibliotecas Cosmos, *scripts* Cosmos e procedimentos de teste
Interface	Uma classe Ruby que sabe como enviar telecomandos e/ou receber telemetrias de um *target*
Configuration Files	Arquivos de configurações de texto simples para definir comandos e pacotes de telemetria e para configurar cada aplicativo Cosmos
Packet Log Files	Arquivos binários contendo comandos registrados ou pacotes de telemetria
Message Log Files	Arquivos de texto contendo mensagens geradas por uma ferramenta Cosmo
Tool	Outro nome dado aos aplicativos Cosmos

Adaptado de BallAerospace (2021e)

target ficam na pasta de configuração do próprio *target* (ex: config/targets/TARGET/screens/tela.txt). Esse arquivos também podem ir para a pasta de telas de destino do sistema (ex: config/targets /SYSTEM/screens/global.txt), quando os arquivos de definição de tela combinam telemetria de vários *target*, mas não é obrigatório (BALLAEROSPACE, 2021g).

Além das palavras chaves utilizadas para a construção das telas, como *SETTING*, que permite configurar a ferramenta ao que foi especificada imediatamente antes dele (ex: SETTING BACKCOLOR 0 0 0, altera a cor do fundo para a cor especifícada em RGB), são utilizadas outras palavras chaves para definir o *layout*, a decoração da tela (linha horizontal título de seção), as ferramentas interativas, as ferramentas de tela (imagens, gifs), configurações padrões, configurações específicas e para apresentar as telemetrias e também seus rótulos na tela de várias formas. Todas essas palavras chaves são melhor descritas em (BALLAEROSPACE, 2021g).

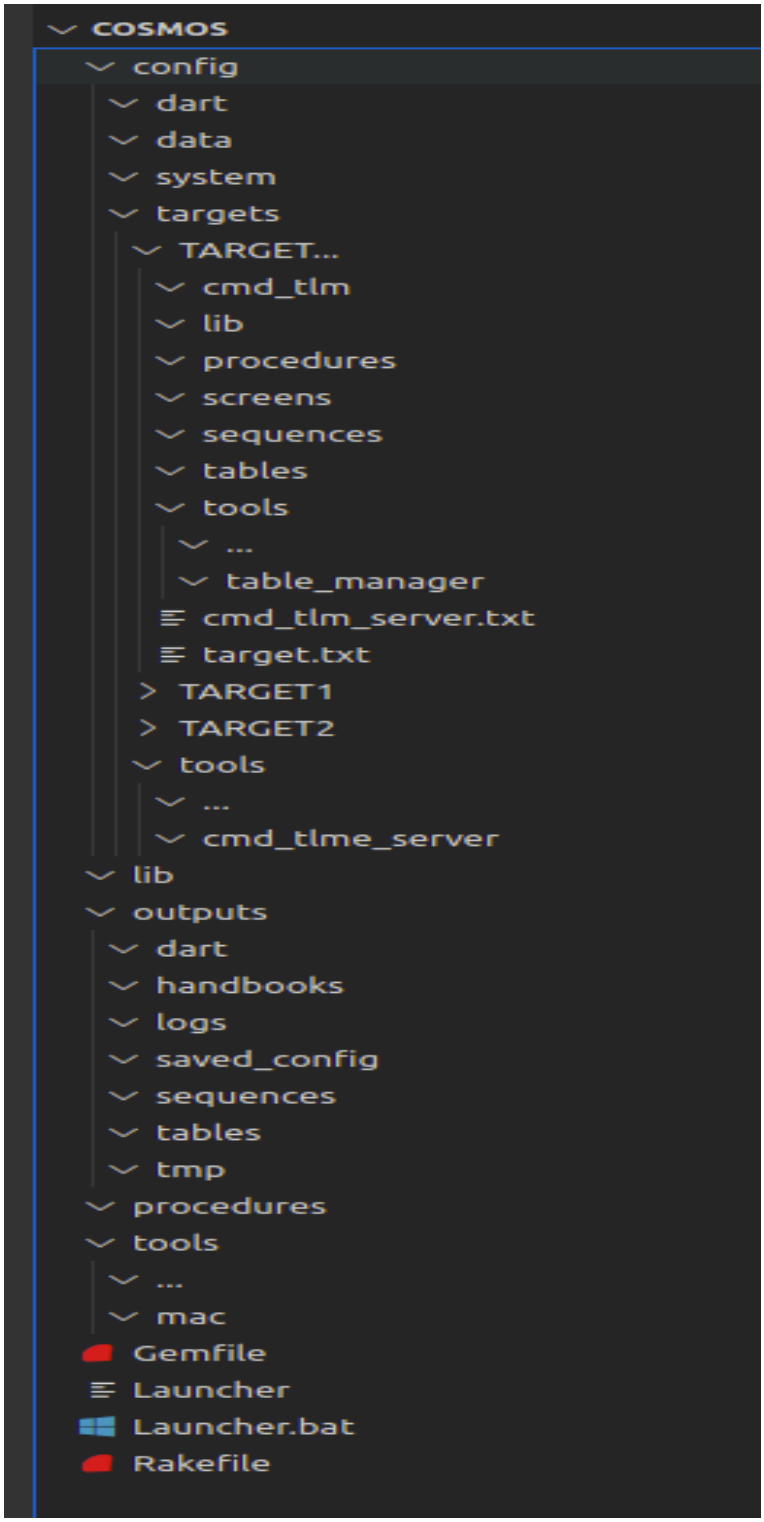


Figura 2.16 – Estrutura de diretório
Fonte: Adaptado de BallAerospace (2021a)

2.5.3 Telecomando

O envio de comandos pode ser feito em forma de *script*, no qual é necessário apenas rodar um algoritmo na linguagem *Ruby*, por meio de uma sequência de comandos enviados manualmente ou fazendo o envio de comandos para cada parâmetro de um alvo. O presente

Quadro 2.3 – Diretórios dentro do Cosmos

Gemfile	Define as gemas e suas versões usadas pela configuração do Cosmos. Se for necessário usar outras joias no projeto, é preciso adicioná-las nesse arquivo e, em seguida, executar *"bundle install"* na linha de comando
config/system	Essa pasta contém o arquivo system.txt, o qual contém configurações comuns a todos os aplicativos, além de definir os *targets* que compõem a configuração do Cosmos
config/targets	Essa pasta contém a configuração de cada *target* que deve ser comandado ou que deve transmitir as telemetrias. As pastas de *targets* devem ser nomeadas com o nome do alvo e todos os caracteres devem estar em caixa alta
config/targets/TARGET	*config/targets/TARGET* é uma pasta que contém as configurações de um *target*
config/targets/ TARGET/cmd_tlm	A pasta *config/targets/TARGET/cmd_tlm* contém arquivos de definições de comando (ex: target_cmds.txt) e de telemetria (ex: target_tlm.txt) para o *target*
config/targets/ TARGET/lib	Contém qualquer código personalizado exigido pelo alvo. Frequentemente, isso inclui uma classe de interface personalizada
config/targets/ TARGET/screens	Contém telas de telemetrias para o *target*
config/targets/ TARGET/tools	Contém arquivos de configuração específicos do alvo
config/targets/TARGET/ cmd_tlm_server.txt	Esse arquivo contém uma parte da configuração para o aplicativo *Command and Telemetry Server* que define como interagir com o alvo específico
config/targets/ TARGET/target.txt	Contém configurações específicas do *target*, como quais classes em Ruby são usadas para a conversão customizadas ou quais parâmetros de comando devem ser ignorados pelo *Command Sender*
config/tools/ cmd_tlm_server	Essa pasta contém o arquivo de configuração para o *Command and Telemetry Server*, por padrão é nomeado como cmd_tlm_server.txt. Esse arquivo define como se conecta a cada alvo

Adaptado de BallAerospace (2021e)

trabalho mostra o envio dos comandos definidos para cada parâmetro por meio do aplicativo *Command Sender*.

O *Command Sender* possibilita o envio de telecomandos do Cosmos para o alvo, ou seja, envia os telecomandos para o *CubeSat*. O envio de comandos é feita de maneira manual, no qual é preciso escolher o *Target* e o tipo de comando a ser enviado. No aplicativo é apresentado os parâmetros dos comandos, os quais estão relacionados a escolha do alvo e do tipo de comando.

Os parâmetros do comando mostrados no aplicativo são: nome, valor ou *status*, unidade e descrição. O nome, a unidade e a descrição são definidos dentro do algoritmo do Cosmos. O controlador do *CubeSat* define o valor ou o *status* a ser enviado, no entanto, o Cosmos possibilita definir um valor padrão para cada parâmetro. O histórico de comandos enviados é mostrado na parte inferior do aplicativo.

3 ALFACRUX

3.1 Contextualização

O Laboratório de Simulação e Controle de Sistemas Aeroespaciais (LODESTAR) da Universidade de Brasília em parceria com a a FAP-DF e a AEB iniciou o projeto AlfaCrux com o objetivo de desenvolver um sistema de comunicação em banda estreita, com a possibilidade de expandir para uma constelação de nanossatélites, com fins educacionais e de demonstração de serviços de comunicação via satélite, indo ao encontro da tendência mundial da indústria espacial (FONSECA, 2020).

O desenvolvimento de satélites de dimensões reduzidas tem a expectativa de levar esse tipo de tecnologia para instituições com menor poder aquisitivo, como as universidades, visando a democratização ao acesso a tecnologias espaciais. Os projetos de missões de pequeno porte vêm crescendo de forma exponencial no Brasil. Sendo assim, o projeto mostra-se como uma tendência mundial. O desenvolvimento em tecnologia, em especial, a aeroespacial é uma iniciativa conjunta com o intuito de revolucionar as iniciativas científicas no Distrito Federal (MCTI, 2020).

É indispensável ressaltar que o projeto do Sistema AlfaCrux, trata-se da primeira missão espacial financiada pelo Governo do Distrito Federal (GDF). Dessa forma, percebe-se a relevância do projeto pioneiro na área que vem desbravando a escassez de investimento na ciência brasileira (CONFAP, 2020).

Por se tratar de um parceria entre o Governo Federal e o Governo do Distrito Federal, o projeto é uma grande oportunidade para impulsionar a educação e o desenvolvimento em Ciência, Tecnologia e Inovação, tanto no DF, quanto em todo o Brasil, além de demonstrar um crescimento do país na área de nanossatélites, e colocar a capital do país na rota de polos de inovação e tecnologia, com a missão de desenvolver áreas, como a telecomunicação e a mobilidade e transformar Brasília na cidade inteligente pioneira da América Latina (VINHOTE, 2020; FAPDF, 2020).

Essa primeira missão espacial é um *CubeSat* 1U com 10 cm de aresta e 1,065 kg que foi lançado em uma órbita heliossíncrona com 500km de altitude no dia 01 de abril de 2022, com vida útil de aproximadamente 3 anos. Essa missão educacional tem como objetivo oferecer uma cobertura confiável de comunicação em banda estreita, demonstrando melhorias na comunicação em lugares remotos, ao implementar serviços de repetidora digital e armazenamento e retransmissão de dados. Neste contexto, permitirá demonstrar melhorias no trabalho de produtores rurais, fornecendo informações, como dados do solo e do clima de regiões remotas (VINHOTE, 2020), dentre outras, (FAPDF, 2020)

O projeto AlfaCrux é composto por uma equipe de 38 indivíduos, sendo 8 docentes, 2 técnicos e 24 discentes da área acadêmica de engenharia da UnB, 2 professores e 2 engenheiros de instituições parceiras, incluindo a coordenadora da missão Serpens-1 do Programa denominado de “Sistema Espacial para Realização de Pesquisa e Experimentos com Nanossatélites"(MCTI, 2020). Vale salientar que a idealização do projeto tem contribuído para a produção de artigos científicos, dissertações de mestrado e teses de doutorado. A difusão do projeto foi em massa, inclusive em mesas redondas, de forma nacional e internacional, mostrando a importância do trabalho para a comunidade e o meio acadêmico envolvido (CONFAP, 2020).

3.2 Sistema AlfaCrux

A Alén Space tem colaborado com o desenvolvimento do Projeto AlfaCrux, permitindo a utilização das suas instalações e do seu programa de formação especializado no desenvolvimento de nanossatélites (*Hands-on Training Programme*). Além disso, o projeto 1U inclui dois sistemas da Alén Space: uma carga útil *TOTEM SDR* (*Sotware Defined Radio*) e uma solução completa do GS-Kit para a instalação de uma estação terrestre de rastreamento de satélite na UnB (ALEN-SPACE, 2021a). O projeto, também conta com subsistemas de energia, painéis solares, baterias, computador de bordo, sistema de comunicações, antenas e estruturas, *software* de controle de bordo e missão (FONSECA, 2020).

O projeto permitirá utilizações inovadoras para a área de comunicação, com o desenvolvimento de soluções para sistemas de comunicação mais eficazes em áreas remotas e isoladas, facilitando a demonstração para a sociedade civil de soluções tecnológicas que permitem comunicação entre dispositivos, podendo ajudar, inclusive, no tratamento remoto contra a Covid-19 em regiões mais isoladas e no combate a disseminação dela e de outras enfermidades ao levar informações mais rapidamente aos locais mais afastados (POVO, 2020; SOARES, 2020).

O sistema AlfaCrux possui três segmentos que interligam entre si (Fig. 3.1): segmento solo, segmento espacial e segmento usuário. O segmento do usuário é composto por um dispositivo de comunicação projetado para a troca de mensagens curtas. O segmento solo inclui todo o equipamento necessário para operar, comandar e controlar o satélite. E o segmento espacial é a plataforma AlfaCrux mais a carga útil *TOTEM SDR*.

Nas subseções abaixo é descrito de forma mais detalhada sobre o segmento espacial, sobre o segmento solo e como é feita a comunicação entre esses dois segmentos. Além disso, é mostrado de forma simplificada as atividades de operações do AlfaCrux. As informações apresentadas nas próximas subseções foram obtidas da documentação interna do projeto AlfaCrux.

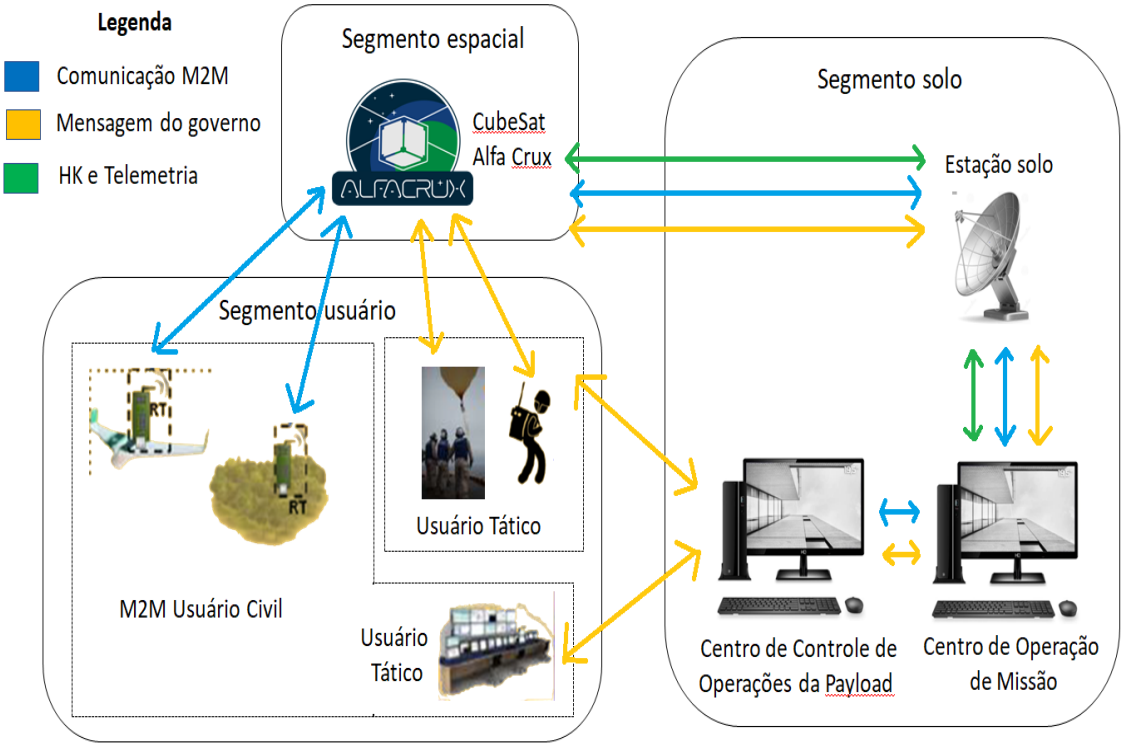


Figura 3.1 – Segmentos do AlfaCrux
Fonte: Adaptado de Reis et al. (2020)

3.3 Satélite AlfaCrux

O segmento espacial tem subsistemas de energia, painéis solares, baterias, computador de bordo, sistema de comunicações, antenas, *software* de controle de bordo e missão e a estrutura. Os painéis solares se localizam nas seis faces do *CubeSat*, ilustrado na Fig. 3.2. As antenas estão em uma face do *CubeSat*, inicialmente fechadas e após 30 minutos do lançamento, elas são abertas.

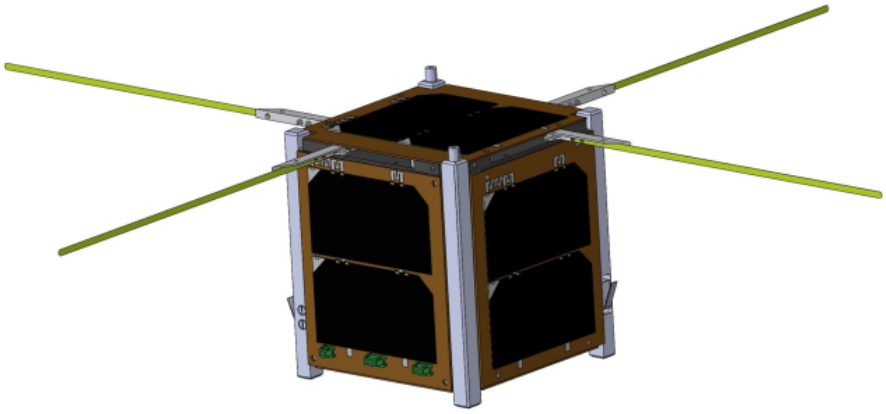


Figura 3.2 – Desenho Técnico do AlfaCrux.
Fonte: AlfaCrux (2022). Não publicado

O sistema de energia, nomeado de EPS (do inglês, *Electric Power System*) é responsável pela captura ou conversão de energia solar em elétrica e pela distribuição de energia para os outros sistemas do AlfaCrux e pelo carregamento da bateria quando apontado para o *CubeSat* estiver iluminado pelo Sol. O conjunto de baterias armazena energia para que possa ser usada durante os trechos de órbita em que o satélite encontra-se em eclipse.

O *TOTEM* é uma *payload* (carga útil), uma plataforma SDR (do inglês, *Software Defined Radio*) de alto desempenho projetado para nanossatélites. O TOTEM opera nas bandas de frequência de nanossatélites mais utilizadas devido um transceptor de ampla faixa de frequência, além de um sistema operacional com Linux embarcado e lógica programável (ALEN-SPACE, 2021c).

No computador de bordo, OBC, tem-se o *software* de controle de bordo e missão. Acoplado ao *hardware* do OBC está o TTC, sistema responsável pela comunicação do AlfaCrux com a estação solo. As definições de protocolo do TTC estão apresentadas na subseção 3.5. A Fig. 3.3 mostra como o EPS, a BATT, o TOTEM e o OBC + TTC estão organizados dentro do *CubeSat*. No Apêndice A é apresentado imagens da vista explodida do projeto real do AlfaCrux.

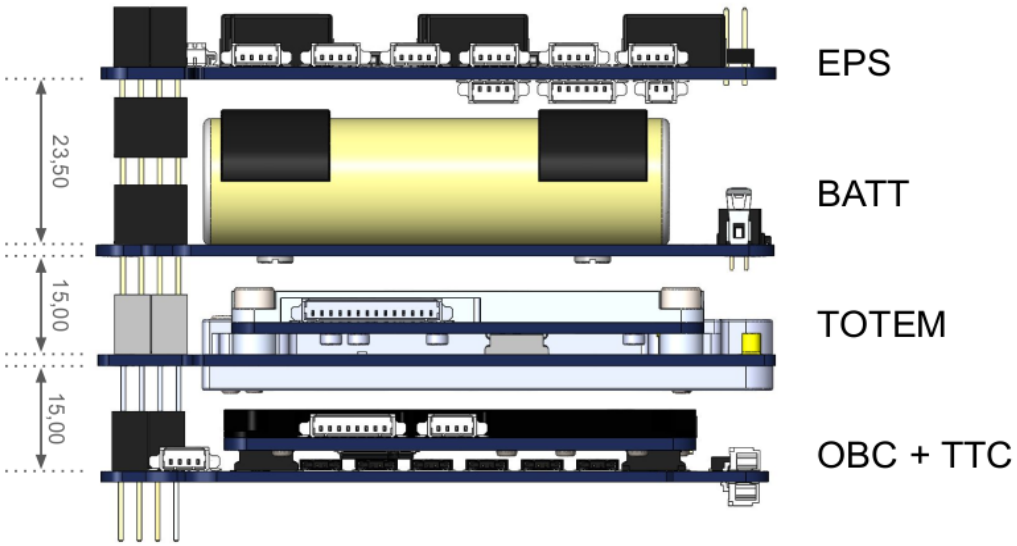


Figura 3.3 – Organização dos sistemas do AlfaCrux.
Fonte: AlfaCrux (2022). Não publicado

O AlfaCrux possui sete modos de operações: *boot-up mode, init mode, startup mode, survival mode, nominal mode, standby mode* e *test mode*. Esses modos podem ser alterados de forma automática ou por telecomando. No dia da finalização do presente trabalho (27 de Abril de 2022) o AlfaCrux estava no modo de segurança (*survivel mode*), ou seja, a operação estava nos primeiros dia de missão. O fluxograma dos modos de operações é apresentando na Fig. 3.4

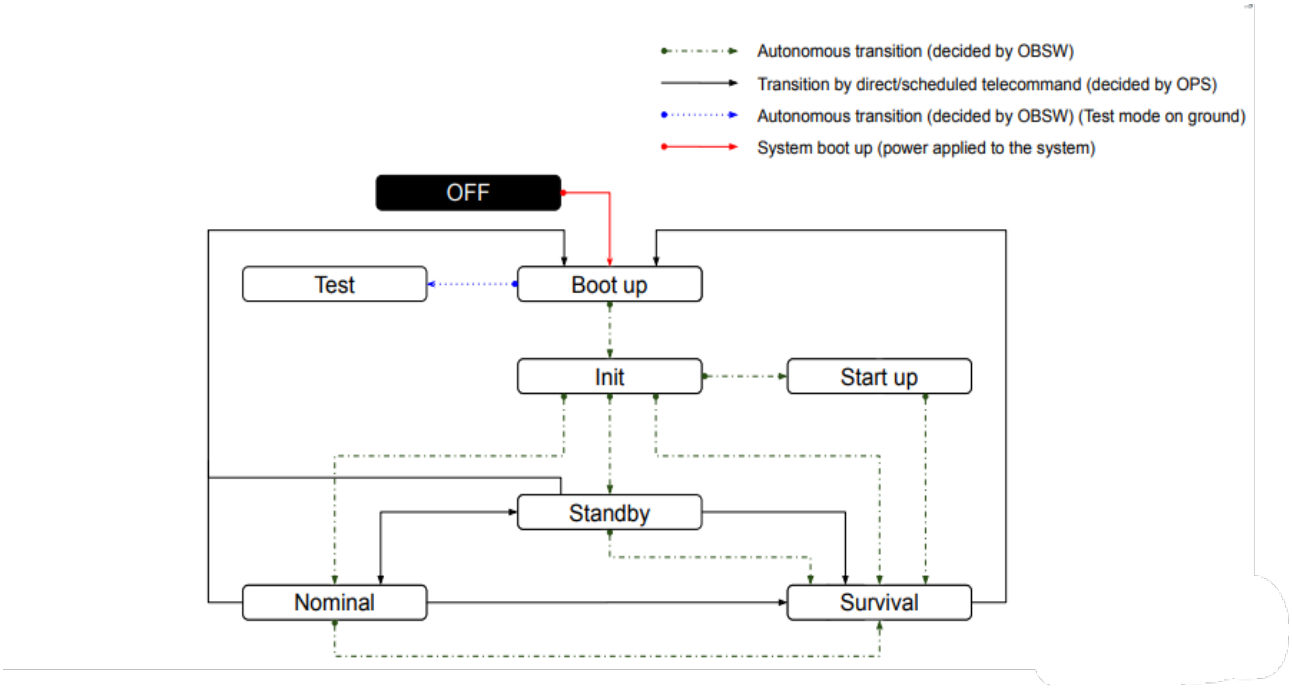


Figura 3.4 – Fluxograma dos modos de operação do AlfaCrux.
Fonte: AlfaCrux (2022). Não publicado

3.4 Segmento Solo

O segmento solo inclui antenas, equipamento de RF e *software* para comunicação UHF (*uplink* e *downlink*) e S (*downlink*) em bandas de radio amadores. O Centro de Operações do Satélite que tem como objetivo monitorar a saúde e os serviços do *CubeSat* durante a vida útil. Assim, é proposto um *software* que permite a recepção, processamento e visualização dos pacotes de telemetria, assim como a gestão, geração e envio de sequências de comando (REIS et al., 2020).

O sistema proposto, apresentando na Figura 3.5, chamaria Sistema de Controle de Missão para o AlfaCrux (SCM αC), faz parte do Centro de Controle de Missão de AlfaCrux e tem comunicação direta com o servidor para o recebimento das telemetrias e o envio dos telecomandos. Imagens da interface segmento espacial - segmento solo, da montagem da antena e da antena montada são mostradas no Apêndice A. Este trabalho visa implementar o SCM αC baseado no *framework* Cosmos.

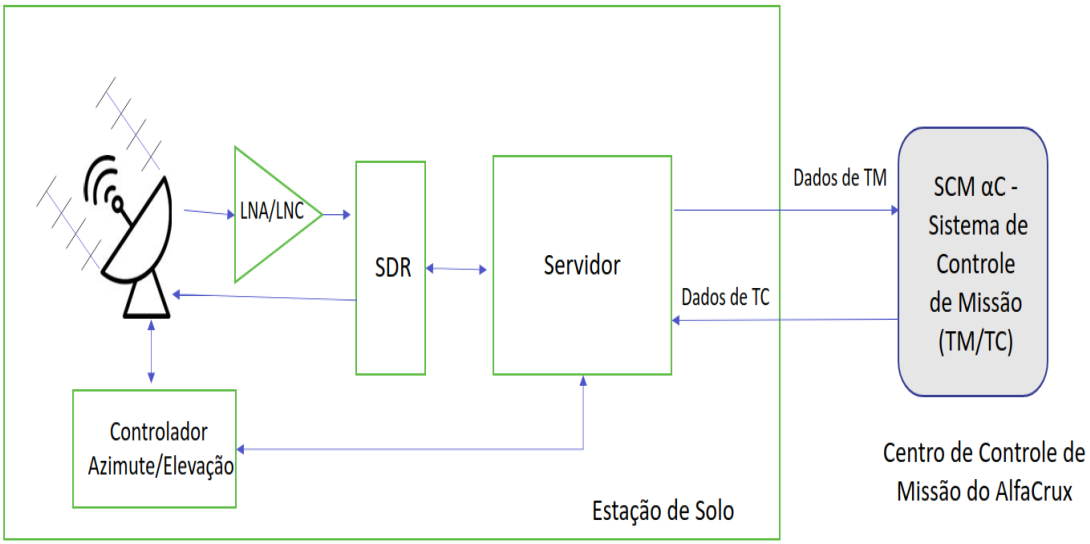


Figura 3.5 – Proposta de um Sistema de Controle de Missão para o AlfaCrux
Fonte: Autor

A Estação Terrestre é uma solução integrada da *Alén Space*, o GS-Kit. Esse kit permite configurar uma estação terrestre e rastrear satélites em órbita terrestre baixa que operam em bandas comerciais e de radioamadores. O kit inclui a estrutura, antenas, cabos, motores de azimutes e elevação, *software* de controle e o SDR-Rack que é um Rádio definido por *software* dual compatível com o Rádio GNU (ALEN-SPACE, 2021b).

3.5 Interface AlfaCrux - Segmento Solo

O AlfaCrux se comunica com a estação solo por meio de frequências UHF fornecidas pela empresa Além Space. Essa comunicação espaço-solo utiliza a interface de camadas da OSI, melhor detalhada pelo CCSDS, descrito na seção 2.4. Os protocolos utilizados em cada camada são mostrados na Fig. 3.6

A camada física utiliza as características da ligação RF, com a frequência TTC de 437,100 MHz tanto para o *uplink* quanto para o *downlink*, comunicação half duplex, quatro níveis de potência de saída de RF: 25 dBm, 27,5 dBm, 28,5 dBm e 30 dBm, modulação GFSK/GMSK e quatro taxas de transmissão: 1200, 2400, 4800 e 9600 bps. A potência de saída e a taxa de bits devem ser ajustados nos valores citados acima de acordo com os dados desejados e para que haja o consumo de potência adequado. A camada de ligação de dados utiliza o modo 5 (ASM+Golay) do equipamento NanoCom AX100 para a sincronização. O AX100 é um transceptor half-duplex de rádio configurável por software projetado para transmissão de longo alcance (GOMSPACE, 2016).

A correção de erro pode ser feita de duas formas distintas. A forma mais avançada é por meio do código Reed-Solomon (RS) fazendo a correção de erros cíclicos não binários

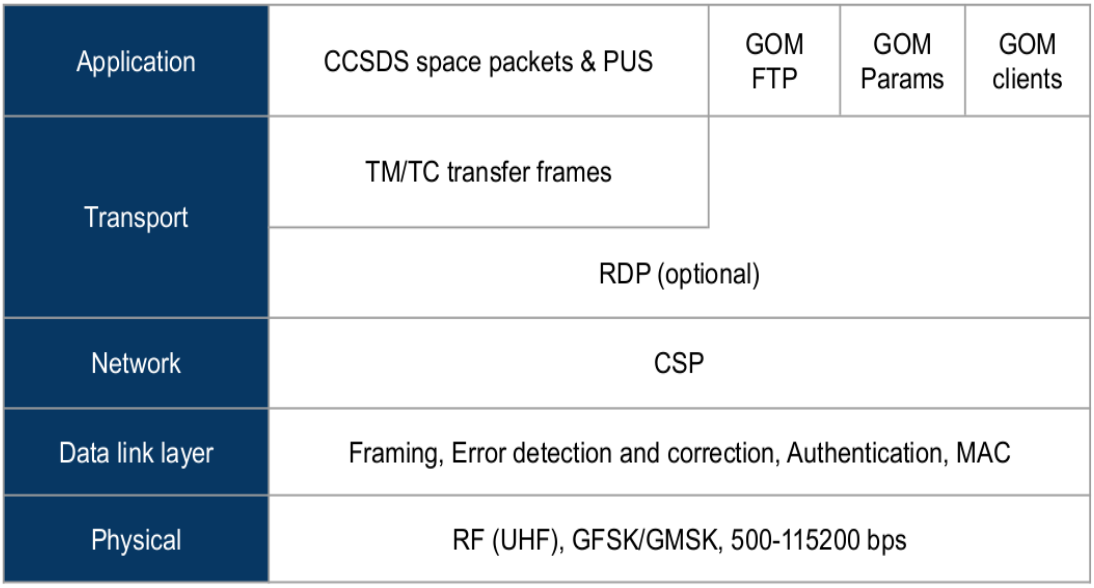


Figura 3.6 – Protocolos das camadas da interface AlfaCrux-solo.
Fonte: AlfaCrux (2022). Não publicado

(SURHONE; TONNOE; HENSSONOW, 2010). A segunda maneira para a detecção de erros é opcional e desabilitada por padrão, o CRC (do inglês, *Cyclic Redundancy Check*). A autenticação das telemetrias e dos telecomando são feitos pelo HMAC (do inglês, *Hash-based Message Authentication Code*). Essa autenticação é do tipo *message authentication code*, contendo uma função *hash* e uma chave para evitar acessos não permitidos ao satélite.

A camada de rede utiliza o protocolo *Cubesat Space Protocol* (CSP). O CSP é um protocolo destinado a camada de redes projetado para *CubeSats*, permitindo que os sistemas integrados distribuídos tenham uma topologia orientada para serviço, facilitando a comunicação. O protocolo CSP segue o modelo TCP/IP e inclui o protocolo de transporte, um protocolo de roteamento e várias interfaces de camada MAC (do inglês, *Medium Access Control*). O protocolo CSP é baseado em um cabeçalho de 32 bits contento tanto a camada de transporte quanto a camada de rede, como mostrado na Fig. 3.7. Esse protocolo é utilizado no AlfaCrux por barramento CAN e I2C e na estação solo pela biblioteca de mensagens assíncronas ZMQ (Fig. 3.8).

A camada de transporte oferece dois protocolos, o RDP (do inglês, *reliable datagram protocol*) e o TMTC *Transfer*, os quais são configurados por telecomandos. O RDP é uma extensão do CSP, seus parâmetros configuráveis são: tamanho de janela, tempo limite de conexão, tempo limite do pacote. O TMTC *Transfer* é baseada nas normas e recomendações CCSDS, permitindo estabelecer uma conexão confiável para mandar os telecomandos e baixar os pacotes de telemetria.

A camada de transferência fornece três tipos de canais: canal bruto, usado para depuração, teste e em operações de contingência, canal não confiável usado para *download* de telemetrias em estações terrestres e para a configuração da ligação, e canal confiável usado para

CSP Header 1.0+

Bit offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	Priority		Source					Destination					Destination Port						Source Port								Reserved				HMAC	XTEA	RDP	CRC
32	Data (0 – 65535 bytes)																															

Figura 3.7 – Protocolo CSP.
Fonte: AlfaCrux (2022). Não publicado

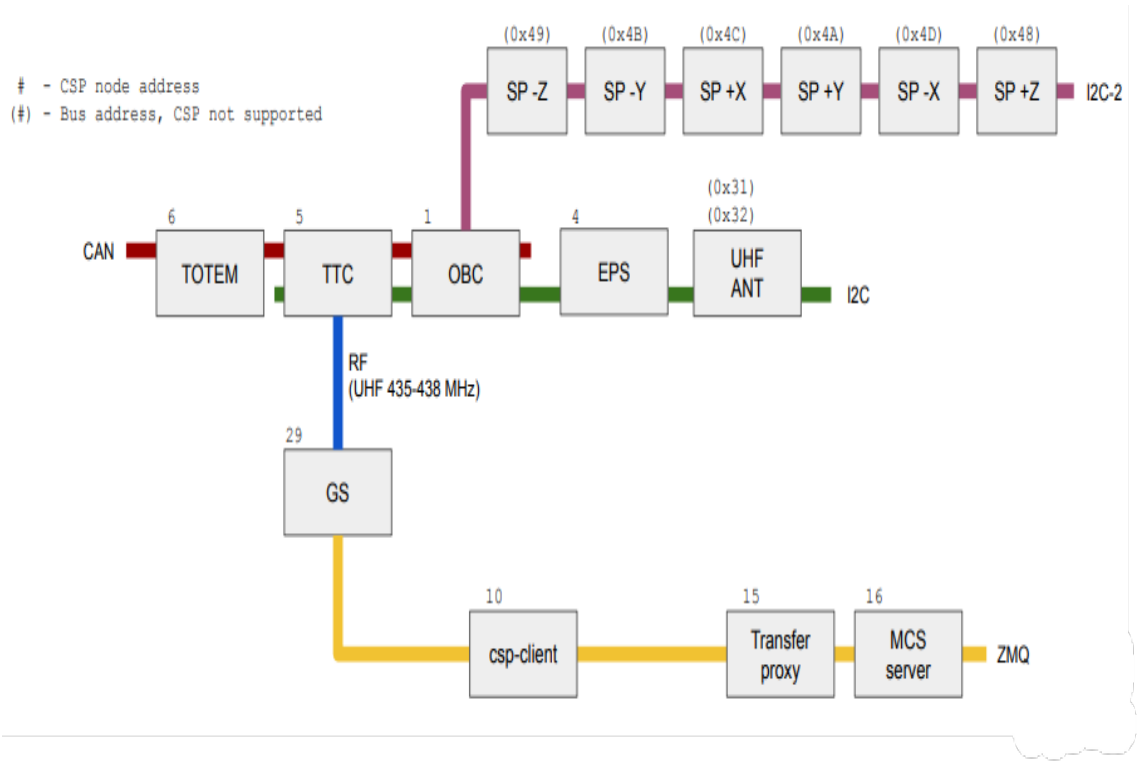


Figura 3.8 – Segmentos do AlfaCrux.
Fonte: AlfaCrux (2022). Não publicado

operações regulares de *uplink* e *downlink*.

A última camada, camada de aplicação, utiliza o CCSDS *space packets* e o PUS *packets*. Os cabeçalhos utilizados no protocolo CCSDS é apresentado na Fig. 3.9, e nas Figs. 3.10 e 3.11 são apresentados os cabeçalhos dos pacotes PUS para telemetria e telecomando, respectivamente.

A comunicação da estação solo com o *CubeSat* tem acessos limitados por dia e tempo limitado em cada acesso. Esses parâmetros de revisita dependem do tipo da órbita em que o satélite foi lançado. Na Tab. 3.1 é apresentando o número de acesso por dia na estação localizada na UnB, a duração do acesso e o tempo de revisita do AlfaCrux, com o mínimo de 10 graus de

packet primary header							packet data field	
packet version number	packet ID			packet sequence control		packet data length	packet secondary header	user data field
	packet type	secondary header flag	application process ID	sequence flags	packet sequence count or packet name			
3 bits	1 bit	1 bit	11 bits	2 bits	14 bits	16 bits	variable	variable
2 octets				2 octets		2 octets	1 to 65536 octets	

Figura 3.9 – Cabeçalho do CCSDS *Space Packet*.
Fonte: AlfaCrux (2022). Não publicado

TM packet PUS version number	spacecraft time reference status	message type ID		message type counter	destination ID	time	spare
		service type ID	message subtype ID				
enumerated (4 bits)	enumerated (4 bits)	enumerated (8 bits)	enumerated (8 bits)	unsigned integer (16 bits)	enumerated (16 bits)	absolute time	fixed-size bit-string
							optional

source data	spare	packet error control
deduced	fixed-size bit-string (deduced)	fixed-size bit-string (16 bits)
	optional	optional

Figura 3.10 – Cabeçalho do pacote PUS - telemetria.
Fonte: AlfaCrux (2022). Não publicado

elevação.

Tabela 3.1 – Tempo de revisita do AlfaCrux teórico com o mínimo de 10° graus de elevação.

GS_UnB	**Mínimo**	**Médio**	**Máximo**
Acesso por dia [-]	2	2,7	3
Duração do acesso [s]	6,8	372,5	472,7
Tempo de revisita [s]	5360,3	32194,7	43147,8

Fonte: AlfaCrux (2022). Não publicado

3.6 Interface Estação Solo - Sistema de Controle de Missão

Após o recebimento das telemetrias na estação solo é necessário processá-las, visualizá-las e armazená-las para, posteriormente, essas informações serem disponibilizadas aos clientes

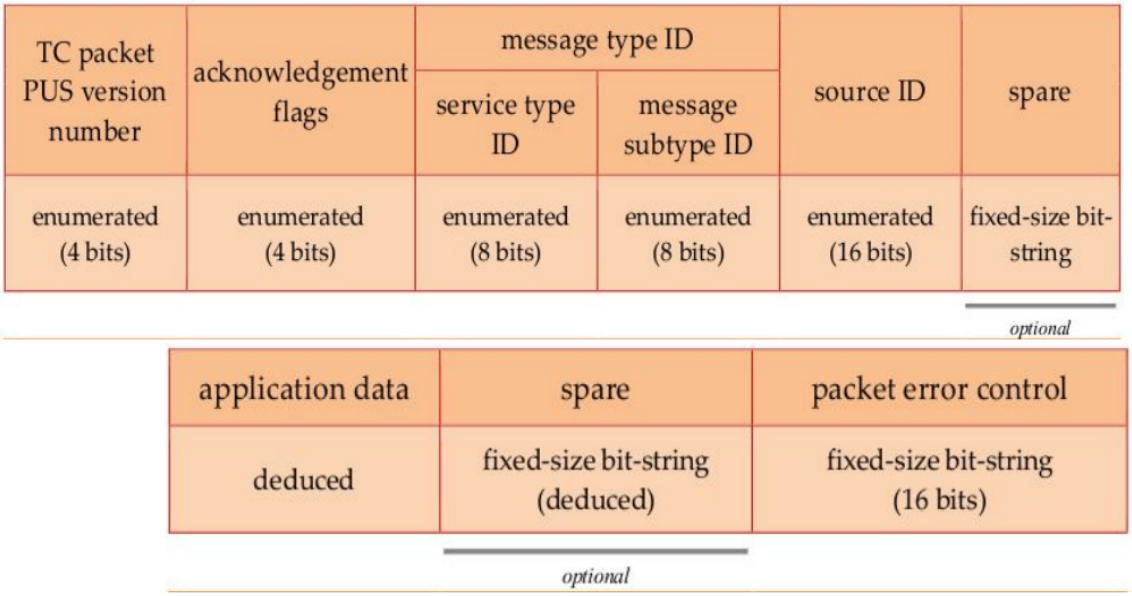


Figura 3.11 – Cabeçalho do pacote PUS - telecomando.
Fonte: AlfaCrux (2022). Não publicado

da missão. Além disso, para os telecomandos serem enviados da estação solo para o *CubeSat* é preciso organizar as rotinas e verificar a data e hora da passagem do AlfaCrux pela estação terrestre.

As atividades citadas acima são realizadas no Sistema de Controle de Missão, a qual se comunica com a estação solo com troca de mensagens utilizando a biblioteca ZMQ (Fig. 3.12). O *Software* de Controle de Missão organiza o recebimento das telemetrias e o envio dos telecomandos. As telemetrias são armazenadas em um banco de dados e disponibilizada para web por meio do serviço API (do inglês, *Application Programming Interface*).

O *Software* de Controle de Missão realiza o procedimento de passagem de rotina. De forma resumida as operações realizadas em uma passagem comum são descritas abaixo. As telemetrias de erro são baixadas primeiro, pois são as telemetrias críticas e caso haja um erro durante a comunicação do AlfaCrux com a estação solo, essas informações não são perdidas.

- Esperar *beacons*;
- Checar o modo do *CubeSat*;
- Enviar um PING e verificar o PONG, se não for recebido, tentar novamente;
- Abrir o canal de comunicação e verificar se o canal confiável de comunicação está aberto, se não tentar novamente;
- Usar o canal e verificar se o canal confiável é usado nos relatórios de TR, se não tentar novamente;
- Iniciar a comunicação;

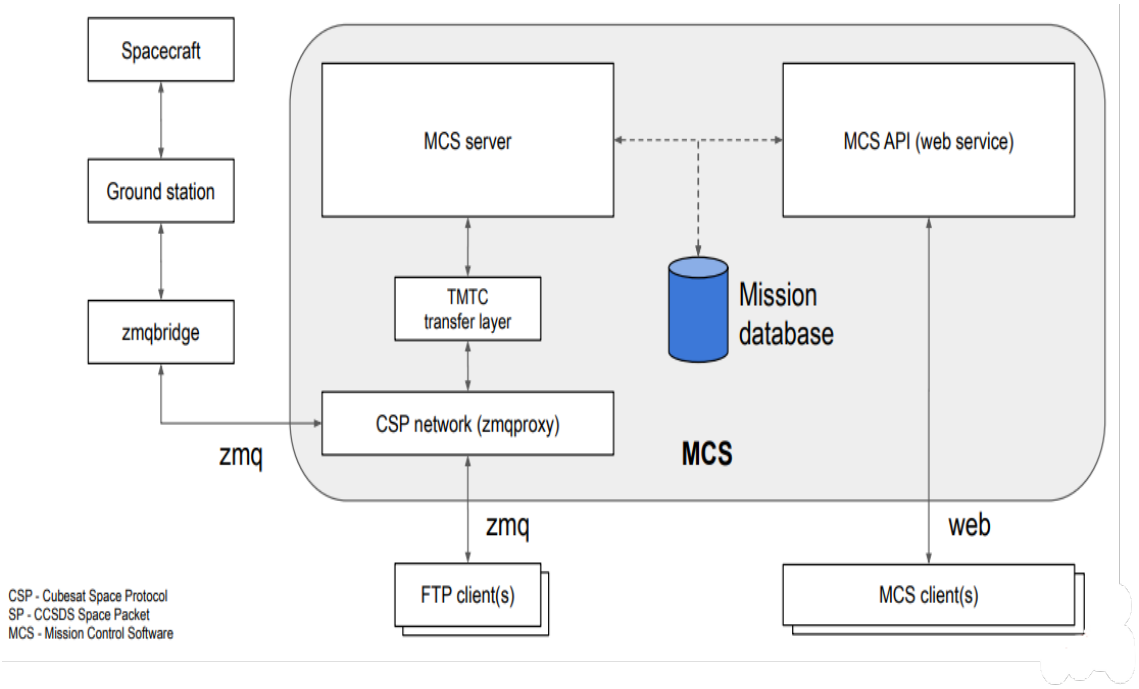


Figura 3.12 – Interface Segmento Solo - Sistema de Controle de Missão.
Fonte: AlfaCrux (2022). Não publicado

- Receber as telemetrias de erro;
- Receber as telemetrias de operações;
- Receber todas as telemetrias;
- Finalizar a comunicação;
- Fechar o canal confiável e verificar se realmente foi fechado;

4 FUNDAÇÃO METODOLÓGICA

A comunicação em tempo real entre o Cosmos e os *targets* é feita por meio do aplicativo *Command and Telemetry Server*, garantindo que todos os comandos e telemetrias sejam registrados (Fig. 4.1). Assim, o primeiro passo é configurar a ferramenta para que possa se conectar ao alvo corretamente.

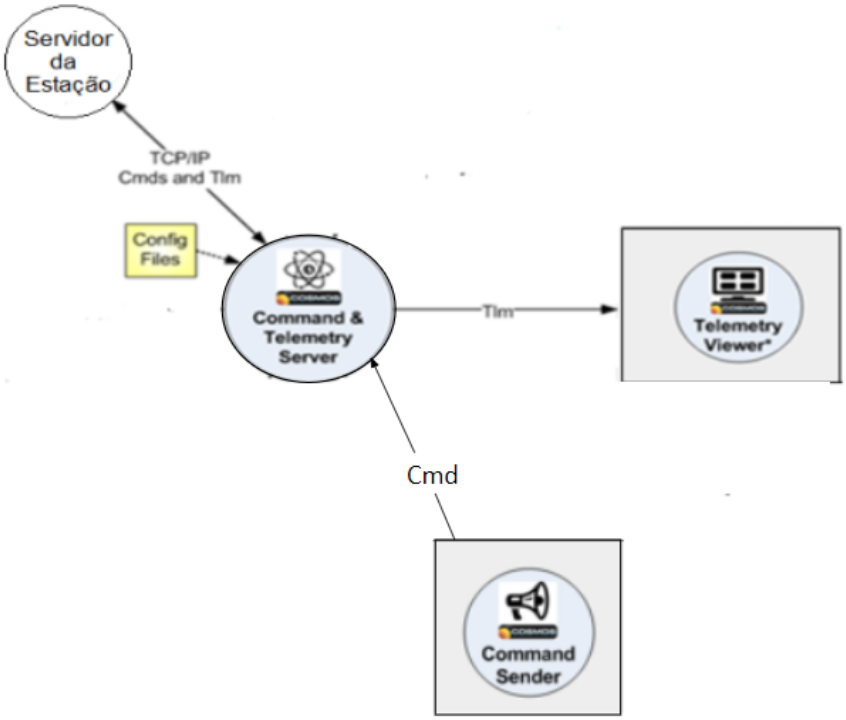


Figura 4.1 – Diagrama das conexões dos aplicativos do Cosmos com o alvo.
Fonte: Adaptado de BallAerospace (2021a)

4.1 Validação do Envio de Dados para o Cosmos

Com o objetivo de validar a comunicação do Servidor da Estação Terrestre com o Cosmos foi criado um programa em C, funcionando como *software* de voo simulado, nomeado genericamente de *CLIENT* ou simulador de TM, que tem a função de enviar dados conhecidos e controlados para o Cosmos, um diagrama dessa conexão pode ser visto na Fig. 4.2.

Na ferramenta Cosmos foi criado um *target*, nomeado de forma genérica de *SERVER*, que recebe as telemetrias do *CLIENT* e as mostras em telas personalizadas, posteriormente, esse *target* será nomeado de acordo com a missão AlfaCrux. Na seção 4.1.1 é explicado como é configurado o Cosmos para se conectar a interface e na subseção 4.1.2 é detalhado o código do *software* simulador de TM.

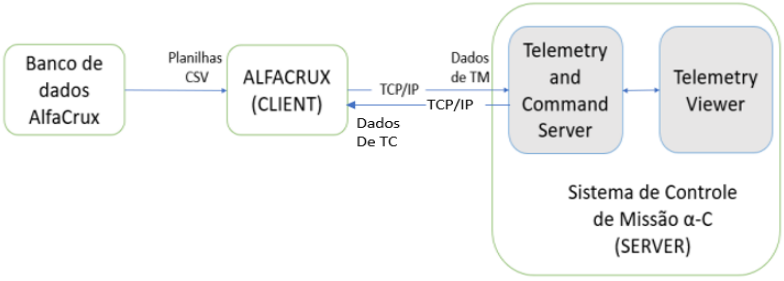


Figura 4.2 – Diagrama das conexões do SCM-αC com o simulador AlfaCrux.
Fonte: Autor

4.1.1 Configuração do *Command and Telemetry Server*

O primeiro passo foi criar uma pasta para o alvo, que enviará as telemetrias, no diretório *config/targets*. O nome dessa pasta precisa ser, obrigatoriamente, com todos os caracteres em caixa alta e optou-se por um nome conciso e que remete ao seu objetivo: *SERVER*. Essa pasta tem todas as informações necessárias para se comunicar com o alvo.

Em seguida, foi criado uma pasta dentro do diretório config/targets/SERVER, nomeada por padrão de cmd_tlm, para colocar o arquivo, server_tlm.txt que define os pacotes de telemetria para o alvo. Também, foi criado, dentro da pasta *SERVER* a pasta *lib* e o arquivo *cmd_tlm_server.txt*. Com o objetivo de informar ao Cosmos que o novo alvo *SERVER* existe, é preciso acrescentar a linha de comando *"DECLARE_TARGET SERVER"* no diretório *config/system/system.txt*. Esse comando diz ao Cosmos para procurar uma pasta chamada *SERVER* em *config/targets*.

No arquivo server_tlm.txt é definido o pacote de telemetria e a própria telemetria. Para definir o pacote de telemetria utiliza-se a palavra chave *TELEMETRY* e em seguida segue-se os parâmetros do Quadro 4.1, resultado em: *"TELEMETRY SERVER TEST_TELEMETRY BIG_ENDIAN "socket communication test - COSMOS as a SERVER*. A telemetria é definida de acordo com seu modificador (ITEM, ARRAY), cada um tem parâmetros diferentes para defini-la. O modificador utilizado para a telemetria do pacote *TEST_TELEMETRY* foi o *APPEND_ITEM*, com a seguinte linha de código *APPEND_ITEM RETURN_TEST 8192 STRING "Test"*, seus parâmetros estão descritos no Quadro 4.2,

A próxima etapa foi configurar o Cosmos para se comunicar com o *CLIENT*, um servidor TCP/IP no endereço 127.0.0.1 usando a porta 65432 para leitura e gravação. Para isso foi preciso criar um *script* em Ruby no diretório config/targets/SERVER/lib. O nome desse *script* não segue resquisitos, assim foi nomeado de *socket_client_intreface.rb* pois a conexão é feita com o algoritmo em C chamado *socket_Client*.

Nesse algoritmo, primeiramente, é chamado o módulo Cosmos com *"module Cosmos"*, uma extensão de todo código que o Cosmos prove. Depois é herdado a classe do servidor TCP/IP padrão do Cosmos com *"class SocketClientInterface < TcpipServerInterface"*, a classe precisa

Quadro 4.1 – Parâmetros para definir um novo pacote de telemetria

Parâmetro	Descrição	Obrigatoriedade
Target	Nome do *target* a que o pacote de telemetria está associada	Sim
Comando	Nome do pacote de telemetria, também, conhecido como mnemônico, sugere-se que seja curto e claro. Deve ser exclusivo para pacotes de telemetria desse *target*	Sim
Endianness	Indica se os dados neste pacote estão no formato Big Endian ou Little Endian	Sim
Descrição	Descrição deste pacote de telemetria, deve ser colocado entre aspas	Não

Adaptado de BallAerospace (2021f)

Quadro 4.2 – Parâmetros para definir o modificador de telemetria *APPEN_ITEM*

Parâmetro	Descrição	Obrigatoriedade
nome	Nome do item de telemetria, deve ser exclusivo no pacote	Sim
Tamanho do bit	Tamanho de bits deste item de telemetria. Valores zero ou negativo podem ser usados para indicar que uma *string* preenche o pacote até o deslocamento do final do pacote especificado por este valor	Sim
Tipo do dado	Tipo de telemetria, sendo válido: INT, UINT, FLOAT, STRING, BLOCK, DERIVED	Sim
Descrição	Descrição deste item de telemetria, deve ser colocado entre aspas	Não
Endianness	Indica se os dados neste pacote estão no formato Big Endian ou Little Endian	Não

Adaptado de BallAerospace (2021f)

ter o nome do arquivo sem espaço e *underline*, com a primeira letra em caixa alta e com a primeira letra que sucede o *underline*, também, em maiúscula. Em seguida, é inicializado a interface com o *CLIENT* na função *initialize*, mostrado no código abaixo, que requer portas de leitura e gravação, o tempo limite de escrita e o tempo limite de leitura, o *"super"* é o método inicializar padrão da classe original e o *"@connect_on_startup = false"* é para o Cosmos não conectar ao alvo na inicialização do *Command and Telemetry Server*. A conexão é feita por outra função nomeada de *connect*, que apenas chama o *super*, função de conexão da classe base, conectando o servidor ao cliente.

```
    def initialize(write_port, read_port, write_timeout, read_timeout,
protocol_type=nil, *protocol_args)
        super
        @connect_on_startup = false
```

```
end
```

Depois, é preciso dizer ao Cosmos como conversar com o *CLIENT*. Essa passo é feito com linhas de códigos na pasta *config/tools/cmd_tlm_servercmd_tlm_server.txt*. A primeira linha inicia-se com a palavra INTERFACE e o restante precisa seguir os parâmetros do Quadro 4.3, a segunda linha diz com qual TARGET é para se conectar e a terceira pede para não reconectar caso a primeira conexão não funcione. Veja a estrutura no código abaixo.

```
INTERFACE SERVER_INT socket_client_interface.rb 65432 65432 nil nil
    TARGET SERVER
    DONT_RECONNECT
```

Quadro 4.3 – Parâmetros da interface de cliente TCP/IP

Parâmetro	Descrição	Obrigatoriedade
Host	Nome da máquina para conectar	Sim
Porta de escrita	Porta para escrever os comandos	Sim
Porta de leitura	Porta a partir da qual ler a telemetria	Sim
Tempo limite de leitura	Segundos para esperar antes de abortar a leitura. Para bloquear na escrita utilize *"nil"*	Sim
Tempo limite de escrita	Segundos para esperar antes de abortar a escrita. Para bloquear na leitura utiliza *"nil"*	Sim
Tipo de protocolo	Ver protocolo	Não
Argumentos de protocolo	Ver protocolo para cada argumento	Não

Adaptado de BallAerospace (2021d)

A Figura 4.3 apresenta o fluxograma dos passos seguidos para fazer a configuração do *Command and Telemetry Server* e a Fig. 4.4 mostra a organização do diretório do *target SERVER*.

4.1.2 Configuração do *Software* Simulador de TM

Esse trabalho utiliza um algoritmo escrito na linguagem C para simular um *software* de voo, com o objetivo de enviar alguns dados semelhantes a dados enviados do *CubeSat* AlfaCrux. Esse algoritmo será utilizado para fins de validar a comunicação do software com o Cosmos, pois os dados enviados são conhecidos e controlados. O *software* é dividido em três algoritmos: *makefile*, *header* e *main*.

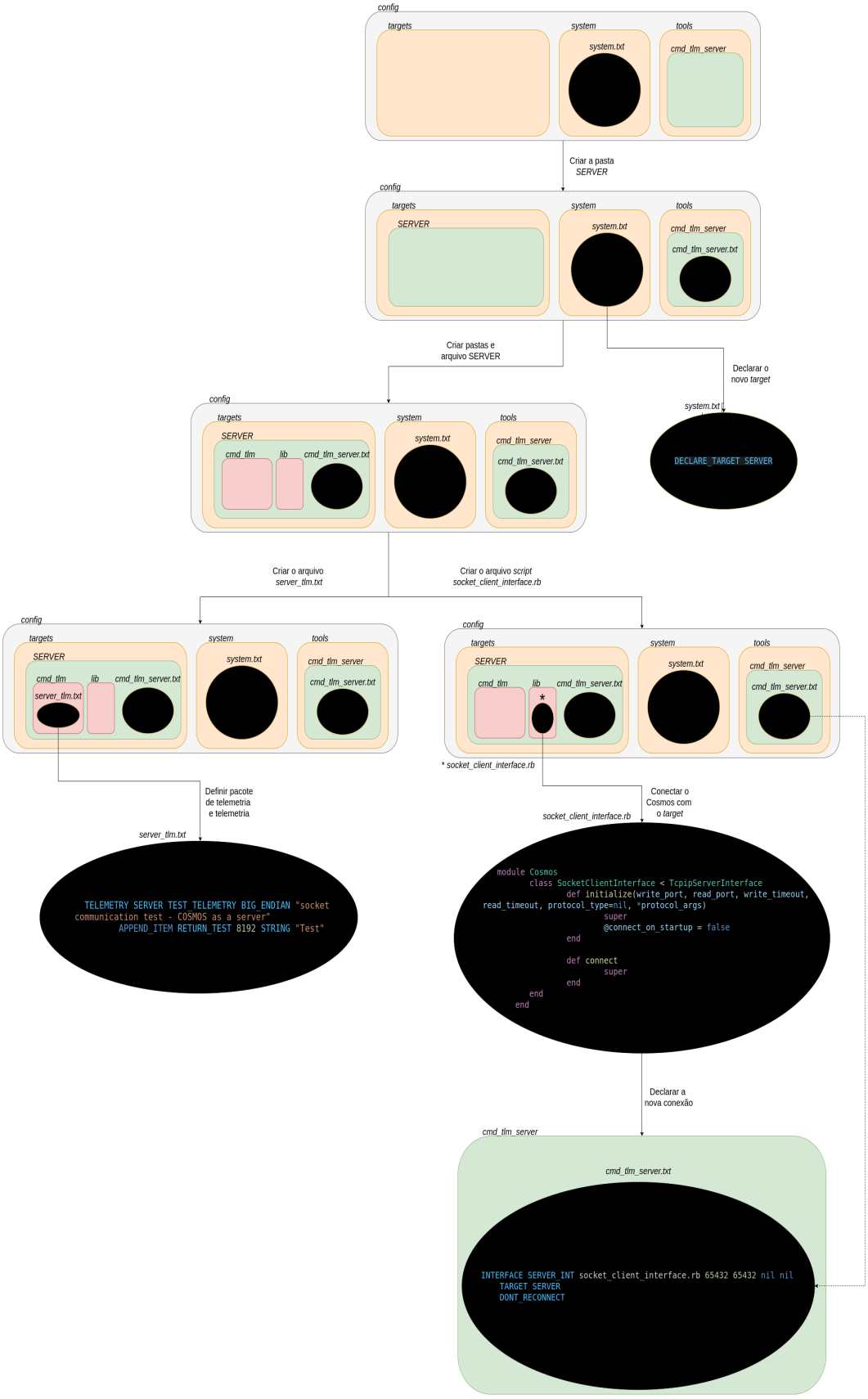


Figura 4.3 – Fluxograma para a construção do *Command and Telemetry Server*
Fonte: Autor

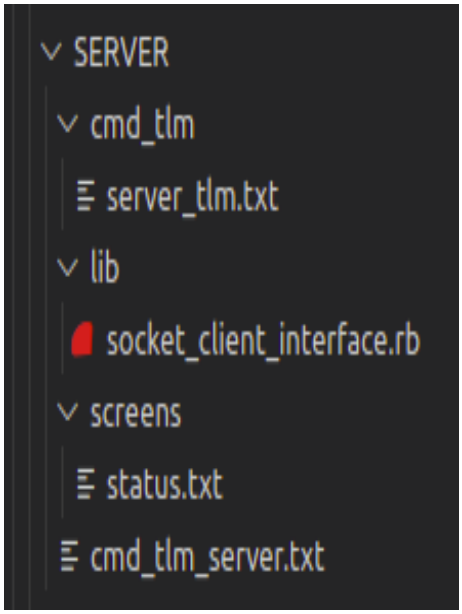


Figura 4.4 – Estrutura de diretório do *target SERVER*
Fonte: Autor

O arquivo *makefile* é utilizado para facilitar a compilação do código em C e fazer a lincagem das bibliotecas utilizadas, além de ser mais portátil. Em suma, esse arquivo transforma os arquivos fontes na extensão .c em arquivos objetos na extensão .o, bem como, lincar os arquivos objetos para transformá-los em um arquivo binário que será executado.

No código, primeiro é definido quatro variáveis: 1)EXECUTABLE, utilizado para criar o arquivo executável na extensão .exe, 2) CC, definido como gcc para compilar os arquivos em C, 3) src, cria uma lista de todos os arquivos .c da pasta e 4) obj, lista que substitui todos os arquivos .c para .o.

Em seguida, são criadas duas receitas: *all* e *clean*, ações que o comando *make* realiza, iniciadas pelo caractere <TAB>. Cada receita é rotulada com um nome diferente para que possa ser chamada no terminal, para isso usa-se *.PHONY*, que tem a estrutura *.PHONY: rótulo*.

A receita *all* cria o arquivo executável, fazendo a compilação dos algoritmos e a ligação das bibliotecas. Essa tem uma dependência de cada um dos arquivos objetos. A receita *clean* apaga os arquivos .c e o executável, para que quando houver mudanças no código em c ou no código *header* haja uma nova compilação desses arquivos, além disso é recomendado limpar os arquivos binários antes de carregar os códigos no GitHub.

No *header*, ou cabeçalho, é incluso bibliotecas e definido as variáveis globais. O *header* construído contém, primeiramente, três *includes* padrões de códigos em C: *stdio.h*, *stdlib.h* e *string.h*. Em seguida, tem os *includes* para trabalhar com *sockets* e algumas definições de tipos de erro.

O código principal é definido pela função *main*. Primeiro é definido as variáveis de descritor de arquivo, endereço, porta, IP, mensagem, mensagem2, tamanho da mensagem, número

de bits recebidos e total do número de bits recebidos. Depois é setado o IP como 127.0.0.1, a porta como 65432 e a primeira mensagem a ser enviada informando que o *socket* foi conectado.

Na função *socket* é passado como parâmetro o sistema, o protocolo que está sendo usado (TCP/IP) e que é IPV4. Depois é construído o endereço do *client*, passando a família, o IP e a porta. A função *connect* estabelece a conexão do *socket* com o seu alvo e a função *send* tenta enviar a mensagem de *echoString* para o alvo, informando que foi conectado. Em seguida são enviados vários valores de inteiros para simular alguma informação do *CubeSat* ou enviado textos personalizados de acordo com que o usuário escreve no terminal. Enfim, o *socket* é fechado e a função termina.

4.1.3 Configuração das Telas de Telemetria

As configurações das telas de telemetria variam de acordo com a necessidade da missão, não tendo um padrão a ser seguido. No entanto, a criação das telas seguem palavras chaves já definidas pelo Cosmos e cada palavra chave tem uma estrutura pré definida. Além disso, a ordem das palavras chaves no código influencia na formatação das telas. O presente trabalho mostra dois exemplos que apresentam algumas funcionalidades dessa *framework*.

Os algoritmos das telas iniciam, sempre, com a palavra chave *SCREEN* que necessita da definição da largura e altura da tela e tempo em segundos entre os *updates* das telas, respectivamente. A largura e altura podem ser definidos como *AUTO*, assim o *Telemetry Viewer* cria o *layout* automático.

No exemplo criado, primeiro foi utilizado um *TABBOOK* para ser criado abas de telas, cada aba é definida com *TABITEM* seguido do nome da aba. A primeira tela (Exemplo1) foi definida /com as seguintes palavras chaves, na ordem apresentada:

- *HORIZONTAL*
  - *VERTICAL*
    - *LABEL*
    - *HORIZONTAL*
      - *LABEL*
      - *VALUE*
    - *LABELVALUE*
    - *LABELVALUE*
    - *LABELVALUE*
    - *TIMEGRAPH*
  - *VERTICAL*
    - *VERICALBOX*

        - *LIMITSBAR*
        - *LABELVALUELIMITSBAR*
    - *HORIZONTALBOX*
        - *LIMITSCOLUMN*
        - *LABELVALUELIMITSCOLUMN*
    - *LINEGRAPH*

Os comandos são intuitivos e a montagem da tela segue uma prioridade, como mostrado na lista acima. O comando *HORIZONTAL* diz que todos os próximos comandos com uma prioridade abaixo (representados pelo símbolo -) serão criados um do lado do outro, dessa forma são criados dois blocos verticais, como mostrado em vermelho na Fig. 4.5a. Nos blocos verticais, todos os comandos com o símbolo * são criados um abaixo do outro, apresentados em verde na Fig. 4.5b.

Dentro do bloco *VERTICAL* a esquerda tem, novamente, a palavra chave *HORIZONTAL*, que faz com que dentro do segundo bloco verde, os comandos *LABEL* e *VALUE* estejam um do lado do outro. O mesmo acontece no bloco vertical a direita, que tem a palavra chave *VERTICALBOX*, deixando os itens de dentro na vertical e o *HORIZONTALBOX*, organizando os itens na horizontal (em laranja na Fig. 4.5c).

A Figura 4.5 é apenas um esboço para representar como o Cosmos entende cada comando e como esses comandos são estruturados. No entanto, as telas não são criadas em estruturas com blocos visíveis e nem organizadas em comprimentos e alturas padronizadas, esses detalhes, bem como a fonte da letra, a cor do fundo e da letra, o tamanho de cada informação, entre outros são definidos com outras palavras chaves como *SETTING*.

O *SETTING* é informado logo depois do comando que ele modificará. Por exemplo, na segunda tela (Exemplo2) tem o comando *LABELVALUE* e na linha de baixo tem o comando *SETTING TEXTCOLOR orange*, o comando *SETTING* deixa a *label* e o valor do *LABELVALUE* em laranja. A construção da tela do Exemplo2 segue a seguinte estrutura:

- *SCROLLWINDOW*
    - *HORIZONTAL*
        - *LABEL*
        - *LABEL*
    - *HORIZONTAL*
        - *LABEL*
        - *VALUE*
    - *LABELVALUE*

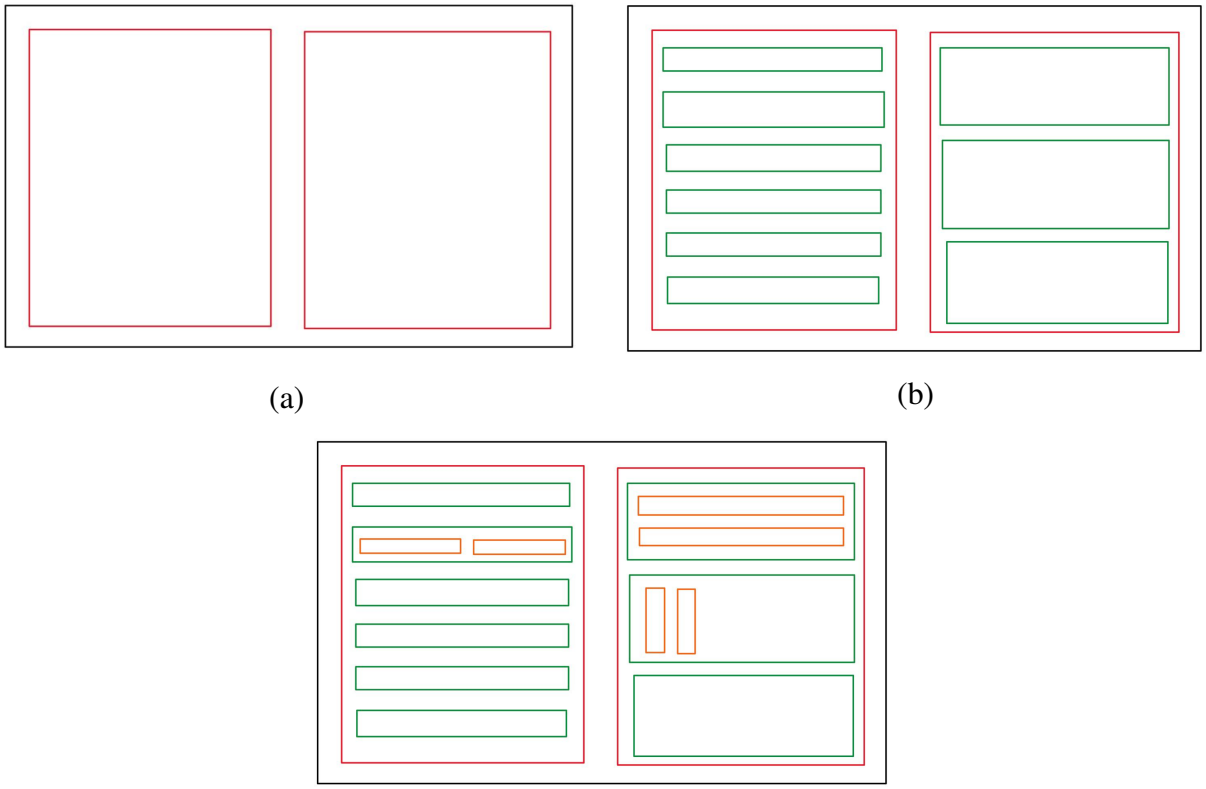


Figura 4.5 – Esboço do desenvolvimento da tela Exemplo1 a) em vermelho: estrutura desenhada dentro de um comando horizontal, b) em verde: estrutura desenhada dentro de um comando vertical e c) em laranja: estrutura desenhada dentro de um comando horizontal

Fonte: autor

- *LABELVALUE*
- *LABELVALUE*
- *TIMEGRAPH*
- *VERTICALBOX*
  - ∗ *LIMITSBAR*
  - ∗ *VALUELIMITSBAR*

  HORIZONTALBOX
  - ∗ *LIMITSCOLUMN*
  - ∗ *VALUELIMITSCOLUMN*
- *CANVAS*
- *SCREENSHOTBUTTON*

O Exemplo2 tem quase os mesmos dados apresentando no Exemplo1, no entanto esses dados são apresentados de forma diferente, além disso, foi acrescentando o comando para que

a tela seja rolável com *SCROLLWINDOW*, como toda a tela precisa ser rolável, o comando precisa ser o primeiro. Foi acrescentado, também, uma imagem com a palavra chave *CANVAS* e um botão para fazer a captura da tela com *SCREENSHOTBUTTON*. Um esboço de como ficaria a tela do Exemplo2 é mostrada na Fig. 4.6.

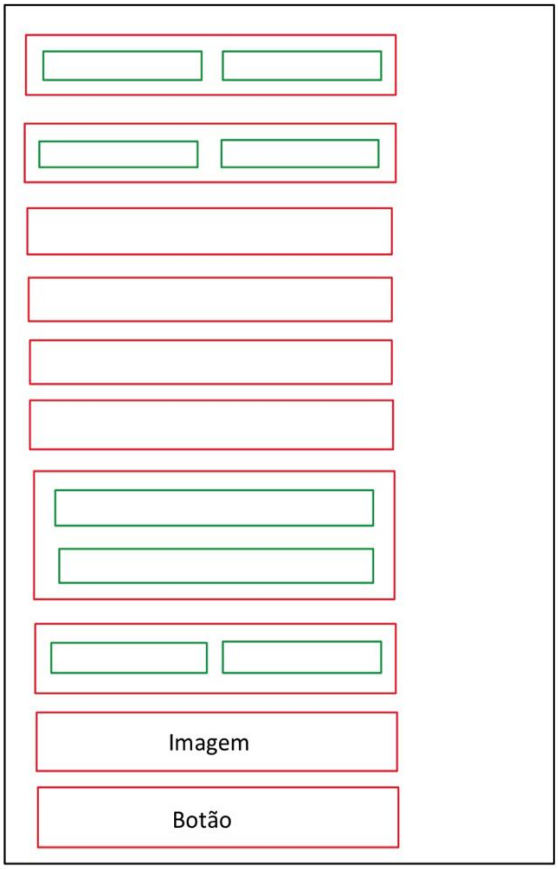


Figura 4.6 – Esboço do desenvolvimento da tela Exemplo2
Fonte: Autor

4.2 Envio dos dados do AlfaCrux para o Cosmos

As telemetrias recebidas pela estação solo são processadas e armazenadas no *Software* de Controle de Missão. A partir desse *Software* são geradas planilhas na extensão csv. Essas planilhas contém todos os dados de telemetrias referente a saúde do AlfaCrux.

O envio das telemetrias do AlfaCrux segue o mesmo princípio descrito para a validação do envio de dados genéricos para o Cosmos. Utiliza-se um código em C, o qual faz a leitura de todos os dados das planilhas e pelo protocolo TCP/IP envia os dados lidos para uma interface do Cosmos, como mostrado no diagrama da Fig. 4.7.

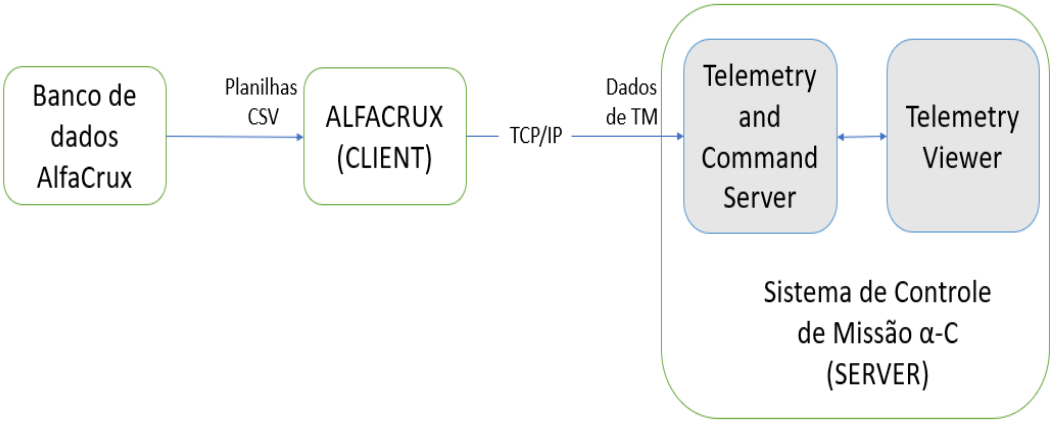


Figura 4.7 – Diagrama das conexões do banco de dados até a visualização de dados no Telemtry Viewer

Fonte: Autor

Tendo em vista que o código em C (SERVER) foi validado na etapa anterior, foi utilizado o mesmo código com a alteração do tipo de dado enviado. A geração de dados conhecidos e controlados foi substituída por dados lidos da planilha csv.

Em relação ao algoritmo do Cosmos, foi criado um alvo nomeado ALFACRUX, seguindo os mesmos passos mostrados para a criação do alvo SERVER. O recebimento das telemetrias foi dividido em quatro pacotes descritos no Quadro 4.4

Quadro 4.4 – Pacotes de Telemetria do AlfaCrux.

Nome do Pacote	**Descrição**
PAINEL_SOLAR	Informações sobre corrente e tensão de entrada dos painéis solares e corrente de entrada do painel solar para a bateria
EPS	Informações sobre corrente de saída do EPS para cada subsistema (OBC, antena, TTC e TOTEM)
TEMP_EPS	Informações sobre as temperaturas internas (conversores, PCB e baterias)
TEMP_PAINEL_SOLAR	Informações sobre as temperaturas externas, ou seja, temperaturas dos painéis solares

Fonte: AlfaCrux (2022). Não publicado

Para cada informação é recebido sete telemetrias críticas as quais contém dados de nome, valor calibrado, valor bruto, se o valor recebido está sendo do limite ou não, tempo em UTC, tempo em UTC SAT e tempo em UTC na GS. Todas as telemetrias são recebidas no formato *STRING* e o número de *bits* depende de cada dado.

4.3 Validação do Envio de Dados do Cosmos para o Alvo

A validação do envio de telecomandos do Cosmos para o alvo é semelhante à validação do envio de telemetria para o Cosmos. Um programa na linguagem C é desenvolvido para simular o alvo AlfaCrux. Os dados recebidos são escritos no terminal para ser verificado se os dados estão sendo enviados pelo Cosmos e recebido pelo simulador corretamente.

O simulador do AlfaCrux segue a mesma estrutura de código do simulador de TM, contendo o *makefile*, *header* e *main*. Visto que a comunicação também é feita pelo protocolo TPC/IP, são utilizadas as mesmas bibliotecas, logo, o *makefile* e o *header* não são alterados e o código principal segue a mesma lógica do código principal do simulador de TM.

No programa principal são definidas as variáveis de descritor de arquivo, endereço, porta, IP, um ponteiro para armazenar a mensagem recebida e uma variável para armazenar o tamanho da mensagem. Em seguida o *script* abre a conexão e fica aguardando uma conexão.

Após estabelecer a conexão do simulador do AlfaCrux com o Cosmos, o simulador do AlfaCrux faz a leitura do *socket* utilizando a função *recv*, armazenando os dados lidos no ponteiro para posteriormente ser mostrado no terminal. Com o objetivo de fechar a conexão de forma segura, uma condição foi implemetada, na qual, se o dado recebido for a *string* "sair", o simulador do AlfaCrux fecha a conexão e finaliza o *script*.

No Cosmos, não é necessário criar outro alvo, o mesmo *target* utilizados para o recebimento de telemetria é utilizado para o envio de telecomando.

Os tipos e os parâmetros do telecomando são criados em um txt dentro da pasta "cmd_tlm", semelhante a criação dos pacotes de telemetria. O txt é nomeado de acordo com o nome do *target*, seguindo o padrão "NomeTarget_cmd.txt

De forma a simular os telecomandos para o AlfaCrux foi criado três tipos de comando: EPS, PAYLOAD e FLAGS. Os tipos de comando EPS e PAYLOAD segue o padrão CCSDS, contendo quatro octetos e o tipo FLAGS tem apenas um *byte* para que possa ser enviado 0 ou 1.

5 RESULTADOS E DISCUSSÕES

5.1 Resultados Genéricos

Após feitas todas as conexões do Cosmos com o *software* de voo simulado, a conexão pode ser estabelecida. Primeiro é preciso rodar o Cosmos com o comando *"ruby cosmos/tools/Launcher"* no terminal. No aplicativo *Command and Telemetry Server* é preciso apertar o botão *Connect* correspondente ao *SERVER_INT*. A conexão com o *target* pode ser estabelecida quando o *Connected?* estiver em *true*. A Figura 5.1 mostra o Cosmos para estabelecer a conexão com o alvo.

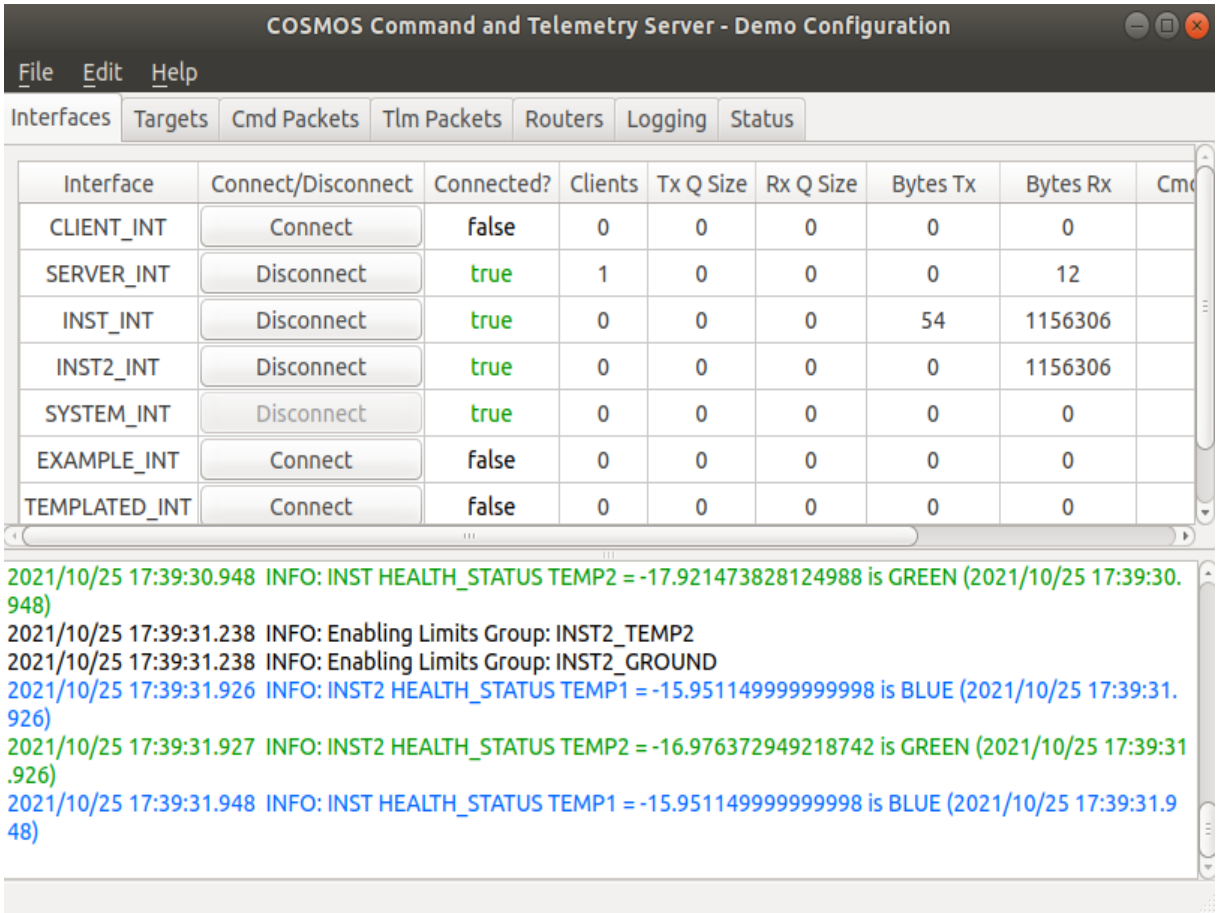


Figura 5.1 – Cosmos conectado para estabelecer a comunicação com o alvo.
Fonte: Autor

A conexão do Cosmos com o alvo é feita definitivamente após rodar o código em C do *software* de voo simulado. No terminal, são feito dois comandos: *"make"* e *"./socketClient.exe"*. Pode-se confirmar a conexão na aba *Tlm Packets* no aplicativo *Command and Telemetry Server*, que aparece a mensagem "Conectado!", como mostrado na Fig. 5.2.

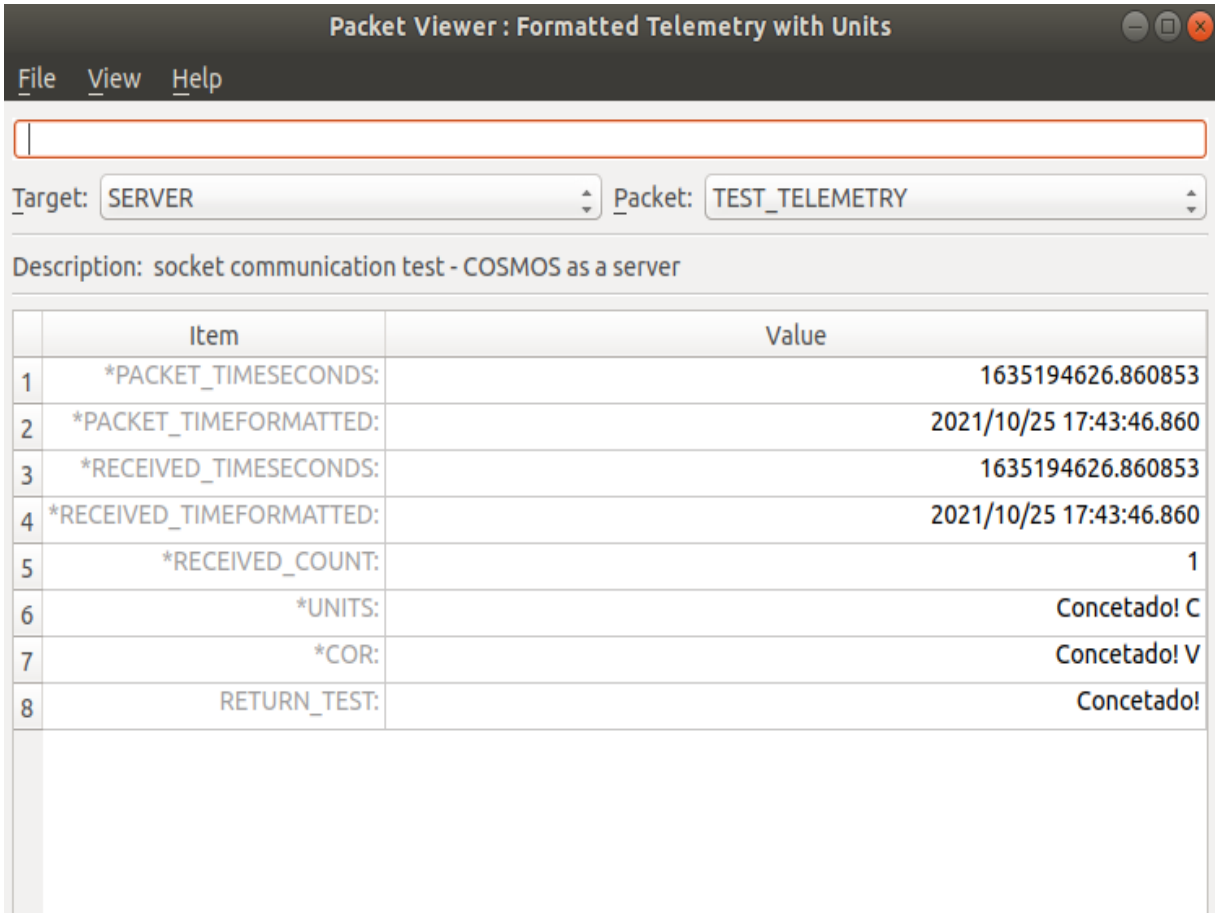


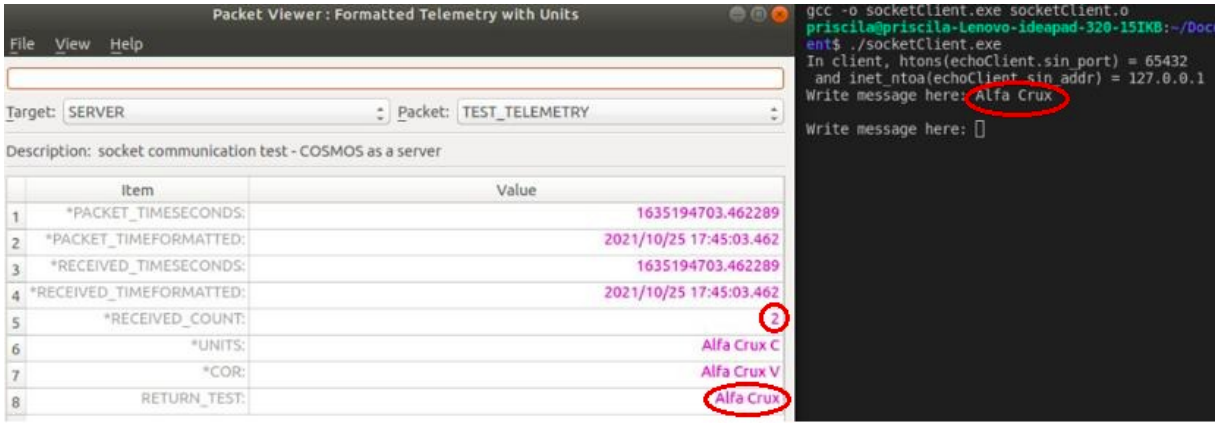
Figura 5.2 – Primeira telemetria enviada, confirmando a conexão do Cosmos com o alvo.
Fonte: Autor

Primeiro, foram enviadas mensagens personalizadas para conferir se a mensagem recebida era idêntica a enviada. A primeira enviada foi "AlfaCrux", mas ela conta como a segunda, já que a primeira foi a mensagem "Conectado!", como pode ser visto na Fig. 5.3a. A segunda, foi "Universidade de Brasília"(Fig. 5.3b) e a terceira, foi "Trabalho de Conclusão de Curso"(Fig. 5.3c).

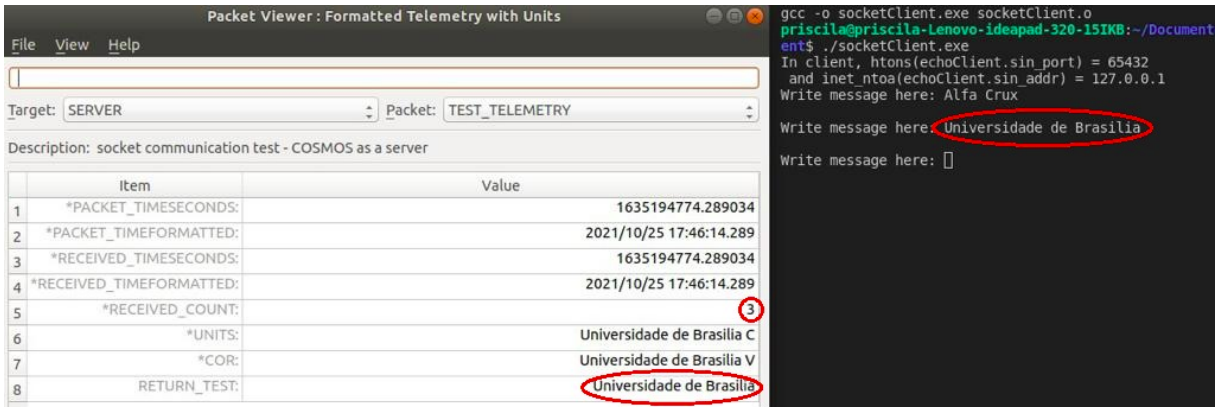
Conferida a comunicação, foram criadas duas telas usando a telemetria, geradas pelo código do *software* de voo simulado, de números inteiros de -10 a 60 e telemetria gerada pelo próprio Cosmos DEMO, representando dados de temperatura. O exemplo 1 (Figura 5.4) mostra como os dados podem ser apresentados: com o valor numérico - podendo mudar a *label*, tamanho, fonte e cor -, em gráficos - telemetria x tempo ou telemetria x amostra - e em barras verticais ou horizontais com limites.

A Figura 5.5 mostra a mesma tela apresentada na Figura 5.4a, no entanto, os blocos que são criados para dividir a tela para cada informação, apresentados no capítulo 4, foram construídos novamente para mostrar onde cada informação se encaixa.

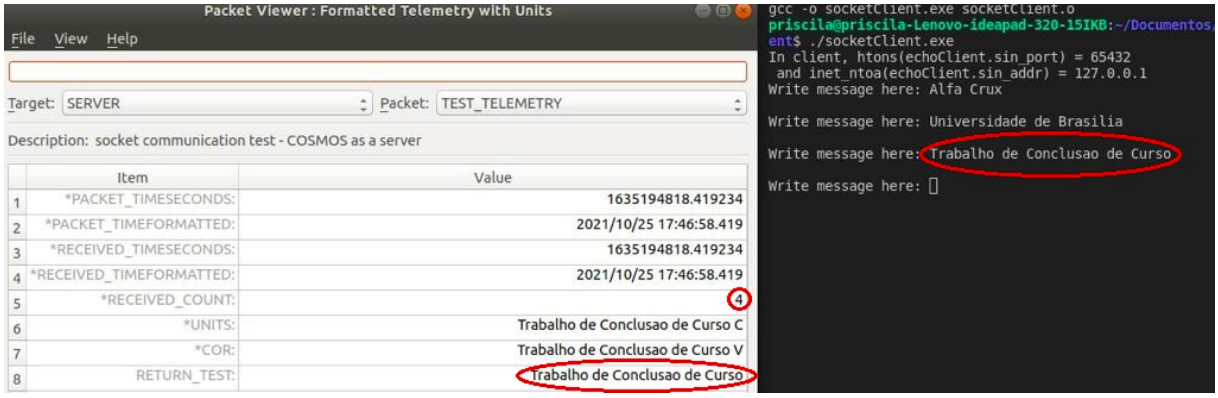
O exemplo 2 (Figura 5.6) mostra como o layout das telas pode ser modificado. É possível mudar a cor de fundo de cada caixa, mudar a cor da letra, colocar imagens, inserir botões e tornar



(a)



(b)



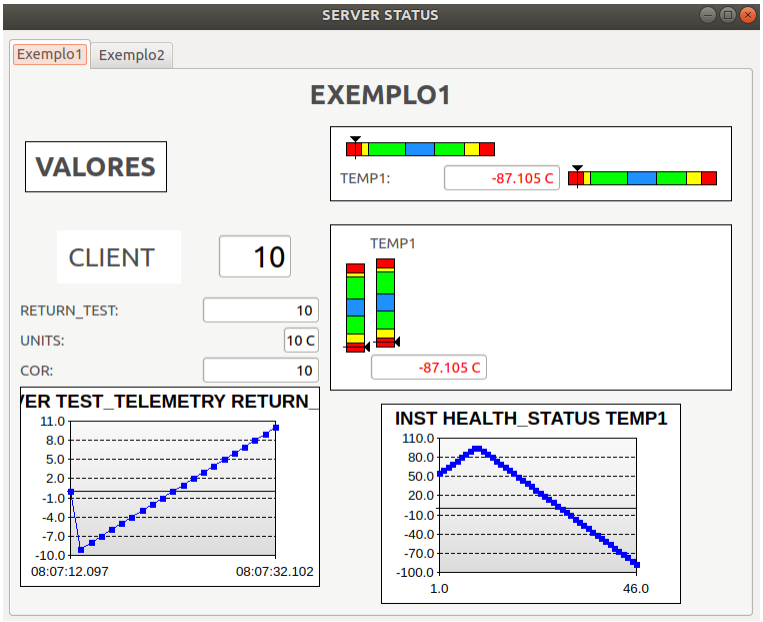
(c)

Figura 5.3 – Envio de telemetria personalizado do alvo para o Cosmos a) mensagem "AlfaCrux", b) mensagem "Universidade de Brasília"e c) mensagem "Trabalho de Conclusão de Curso.

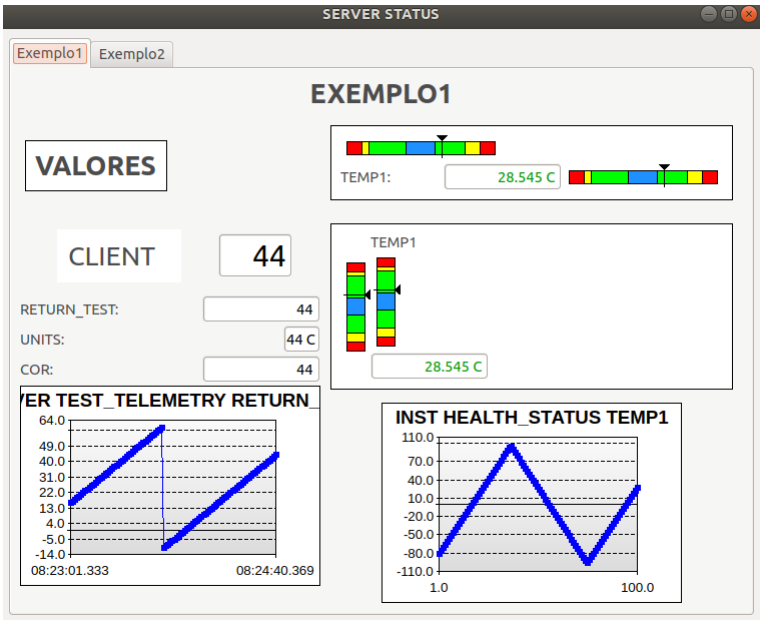
Fonte: Autor

a tela rolável. Além disso, as telas podem ser colocadas em abas, como pode ser observado nas Figs 5.4 e 5.6, o Exemplo 1 e o Exemplo 2 estão na mesma tela, no entanto, em abas diferentes.

Observe que a tela do Exemplo 2 não coube em apenas uma tela, fazendo ser necessário a barra para rolar. A cor do texto do *"RETURN_TEST"*, da *"UNITS"* e da "cor"foram alterados e a cor de fundo dos blocos que mostra as barras de limites, também, foram modificados. Foi



(a)

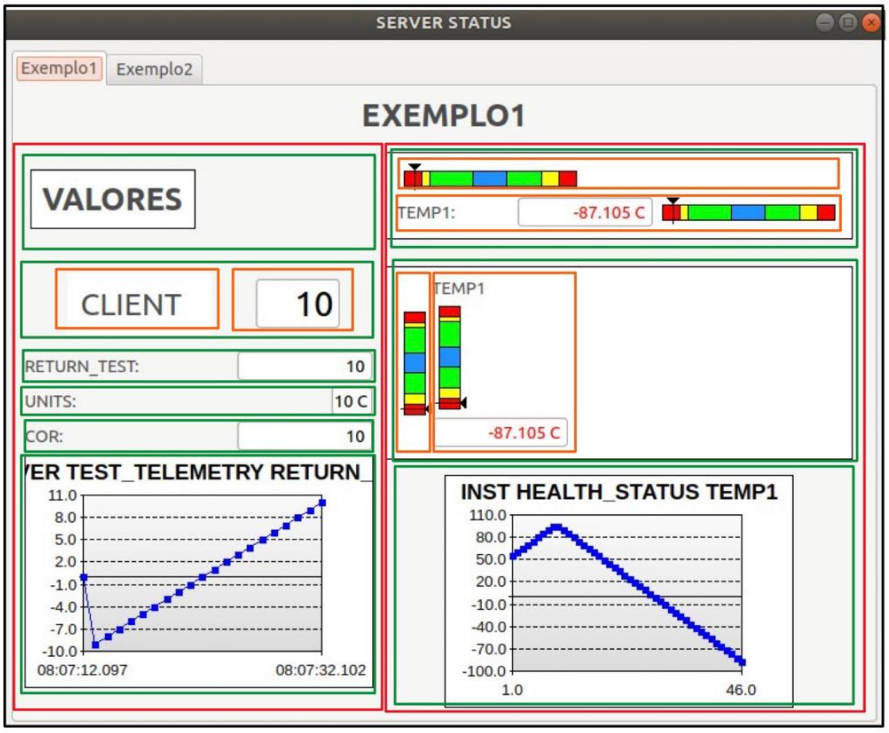


(b)

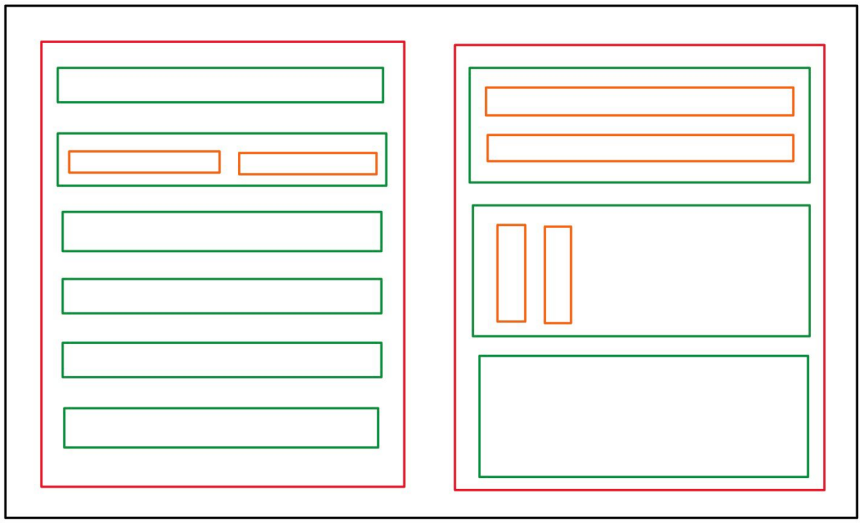
Figura 5.4 – Tela de telemetria - Exemplo 1.
Fonte: Autor

acrescentado a logo da UnB e o botão para fazer a captura de tela.

Note que os dados estão mais organizados na tela do Exemplo1 em comparação com os dados do Exemplo2, além disso, não é preciso ficar modificando a tela para visualizar todos os dados. Esses detalhes de organização da tela, cor e fonte da letra torna a visualização dos dados mais fácil e intuitiva.



(a)



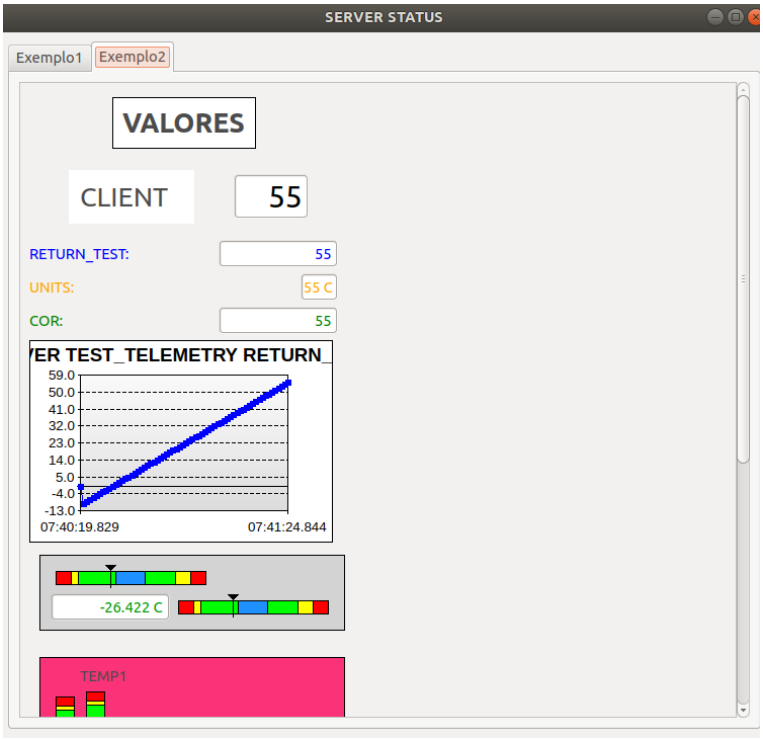
(b)

Figura 5.5 – a) Tela de telemetria - Exemplo1 com os blocos desenhados por cima e b) blocos da tela do Exemplo 1.

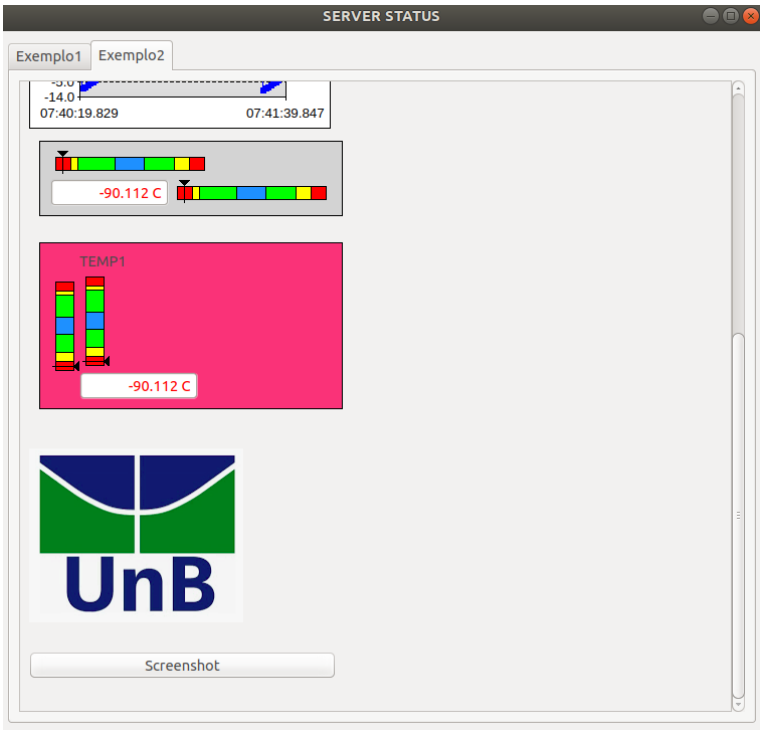
Fonte: Autor

5.2 Resultados para o AlfaCrux

O recebimento dos dados do AlfaCrux, segue a mesma metodologia mostrada na seção acima. Primeiro é preciso habilitar a interface AlfaCrux, como mostrado na Fig. 5.7, para que o servidor possa conectar com o Cosmos. Feita a conexão do Cosmos com o servidor do AlfaCrux, inicia-se a transmissão das telemetrias.



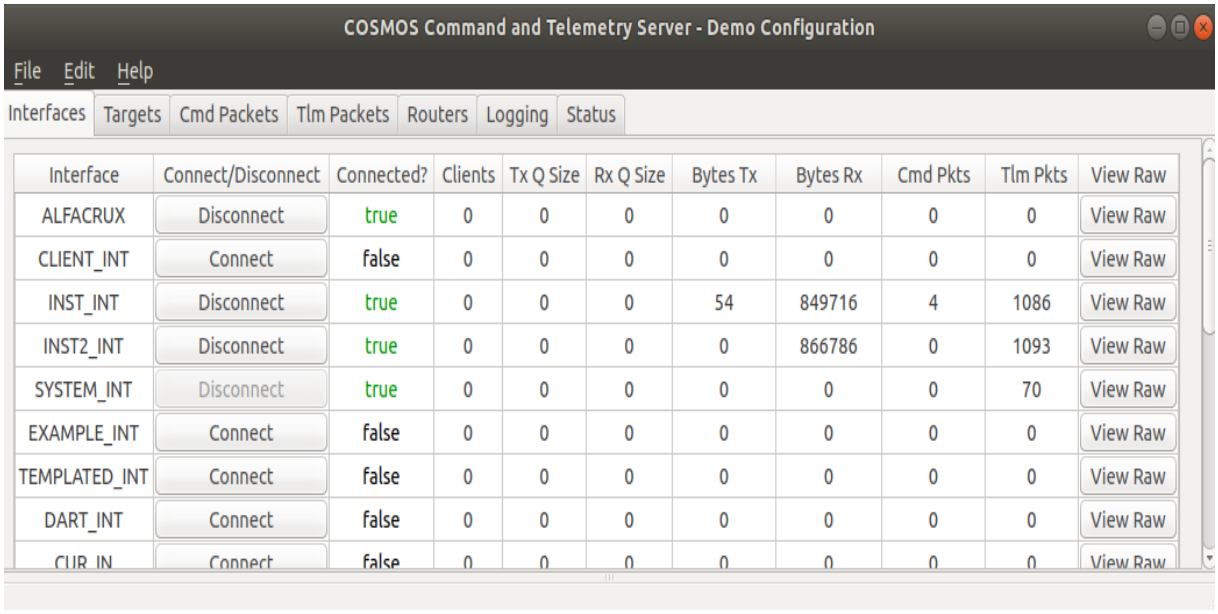
(a)



(b)

Figura 5.6 – Tela de telemetria - Exemplo 2.
Fonte: Autor

O servidor faz a leitura das planilhas contendo os dados do AlfaCrux, adquiridos pelo laboratório LODESTAR e envia para o Cosmos por meio da interface TCP/IP. As telemetrias recebidas são separadas em quatro pacotes, descritos no Quadro 4.4.



Interface	Connect/Disconnect	Connected?	Clients	Tx Q Size	Rx Q Size	Bytes Tx	Bytes Rx	Cmd Pkts	Tlm Pkts	View Raw
ALFACRUX	Disconnect	true	0	0	0	0	0	0	0	View Raw
CLIENT_INT	Connect	false	0	0	0	0	0	0	0	View Raw
INST_INT	Disconnect	true	0	0	0	54	849716	4	1086	View Raw
INST2_INT	Disconnect	true	0	0	0	0	866786	0	1093	View Raw
SYSTEM_INT	Disconnect	true	0	0	0	0	0	0	70	View Raw
EXAMPLE_INT	Connect	false	0	0	0	0	0	0	0	View Raw
TEMPLATED_INT	Connect	false	0	0	0	0	0	0	0	View Raw
DART_INT	Connect	false	0	0	0	0	0	0	0	View Raw
CUR_IN	Connect	false	0	0	0	0	0	0	0	View Raw

Figura 5.7 – Cosmos conectado para estabelecer a comunicação com o servidor ALFACRUX.
Fonte: Autor

O pacote PAINEL_SOLAR contém as informações de tensão e corrente nos painéis solares em todos os eixos (+/-X, +/-Y e +/-Z) e a corrente de entrada do painel solar para a bateria. A tela criada para a visualização desses dados é mostrada na Fig. 5.8.

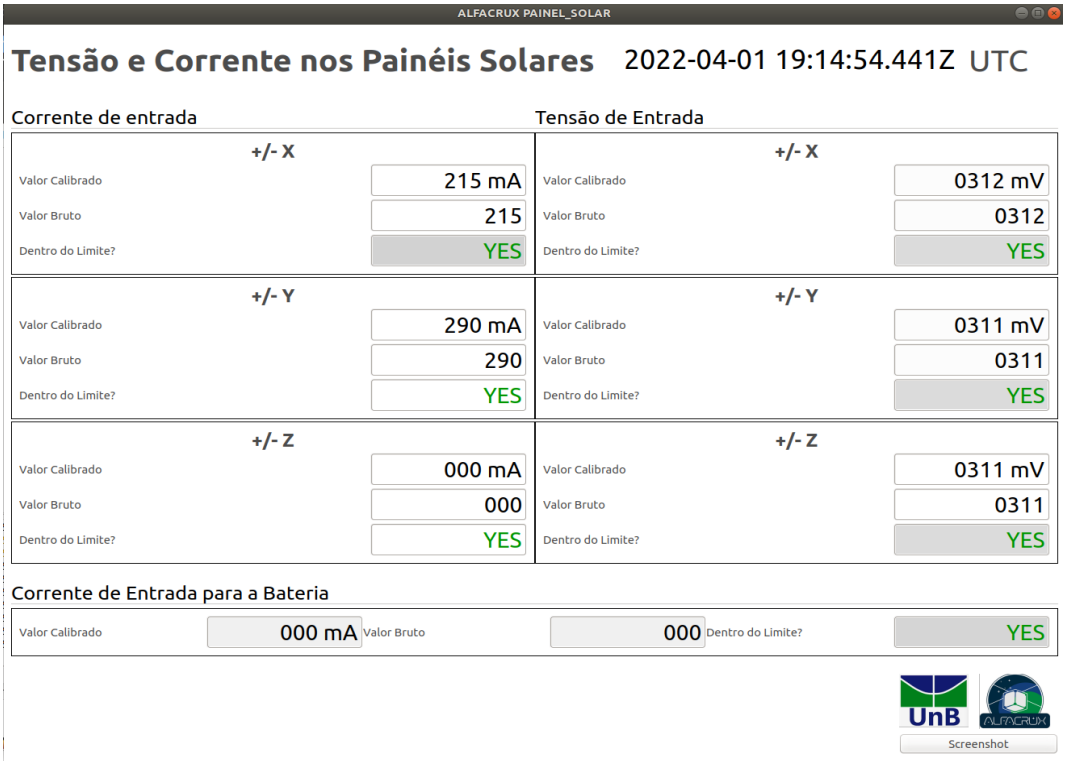


Figura 5.8 – Visualização das telemetrias do pacote PAINEL_SOLAR.
Fonte: Autor

O pacote EPS recebe as informações da corrente de saída do EPS para os subsistemas do AlfaCrux. De forma a padronizar as telemetrias, o início do nome indica a qual telemetria

se refere e o final do nome da telemetria indica qual informação é apresentando. Por exemplo: CUROUT_0_RAW_VALUE, essa telemetria mostra o valor bruto da corrente de saída para o OBC. A Tabela 5.1 mostra quais são as telemetrias e qual a informação de cada telemetria. Para a visualização desses dados foi criado a tela apresentada na Fig. 5.9.

Tabela 5.1 – Nome e informação das telemetrias relacionadas a corrente de saída do EPS

Nome da Telemetria	Informação da telemetria
CUROUT_0	Corrente de saída para o OBC
CUROUT_1	Corrente de saída para a antena
CUROUT_2	Não conectado
CUROUT_3	Corrente de saída para o TTC
CUROUT_4	Não conectado
CUROUT_5	Corrente de saída para o TOTEM

Fonte: AlfaCrux (2022). Não publicado

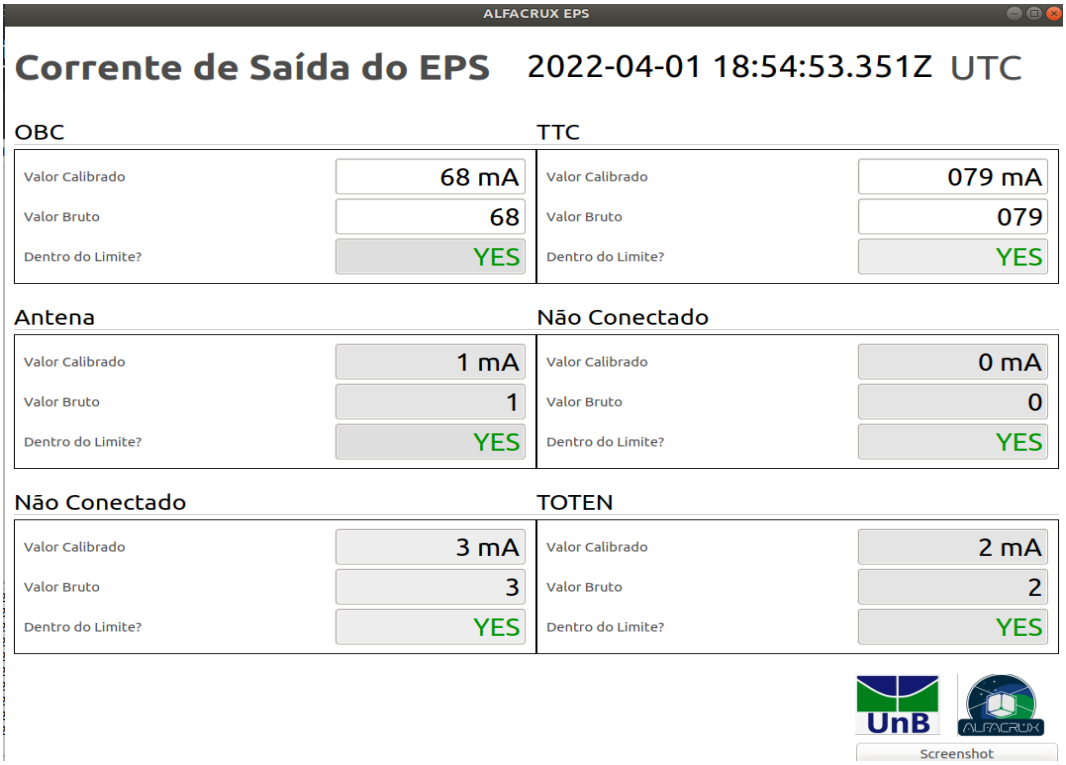


Figura 5.9 – Visualização das telemetrias do pacote EPS.
Fonte: Autor

O pacote TEMP_EPS contém as telemetrias relacionadas as temperaturas internas do AlfaCrux e o pacote TEMP_PAINEL_SOLAR recebe as dados das temperaturas externas, ou seja, as temperaturas dos painéis solares. Esses dois pacotes também seguem a mesma estrutura do pacote EPS, detalhados na Tab. 5.2 e na Tab. 5.3. A visualização desses dados é feita em duas telas separadas, como mostrados nas Figs. 5.10 e 5.11.

Tabela 5.2 – Nome e informação das telemetrias relacionadas a temperaturas internas do AlfaCrux

Nome da Telemetria	Informação da telemetria
TEMP_0	Temperatura EPS - Converso Boost 0
TEMP_1	Temperatura EPS - Converso Boost 1
TEMP_2	Temperatura EPS - Converso Boost 2
TEMP_3	Temperatura EPS - PCB
TEMP_4	Temperatura EPS - Bateria - Par 1
TEMP_5	Temperatura EPS - Bateria - Par 2

Fonte: AlfaCrux (2022). Não publicado

Tabela 5.3 – Nome e informação das telemetrias relacionadas a temperaturas externas do AlfaCrux

Nome da Telemetria	Informação da telemetria
TEMP_NX	Temperatura do painel solar -X
TEMP_PX	Temperatura do painel solar +X
TEMP_XY	Temperatura do painel solar -Y
TEMP_PY	Temperatura do painel solar +Y
TEMP_XZ	Temperatura do painel solar -Z
TEMP_PZ	Temperatura do painel solar +Z

Fonte: AlfaCrux (2022). Não publicado

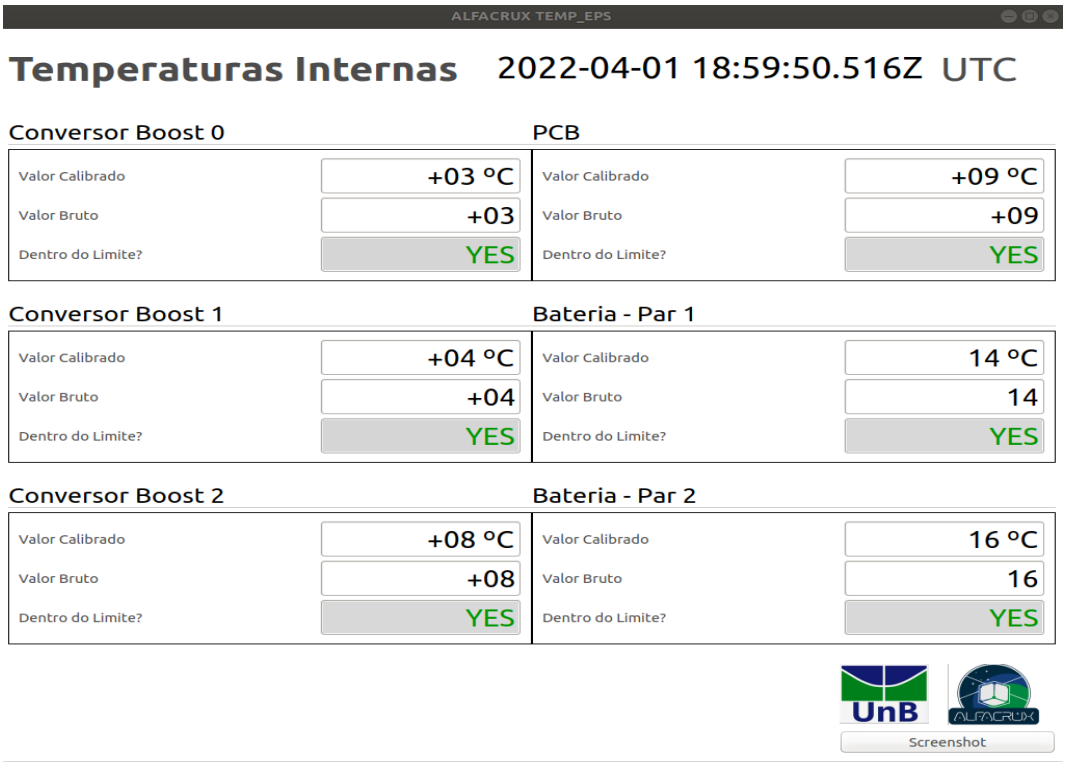


Figura 5.10 – Visualização das telemetrias do pacote TEMP_EPS.
Fonte: Autor

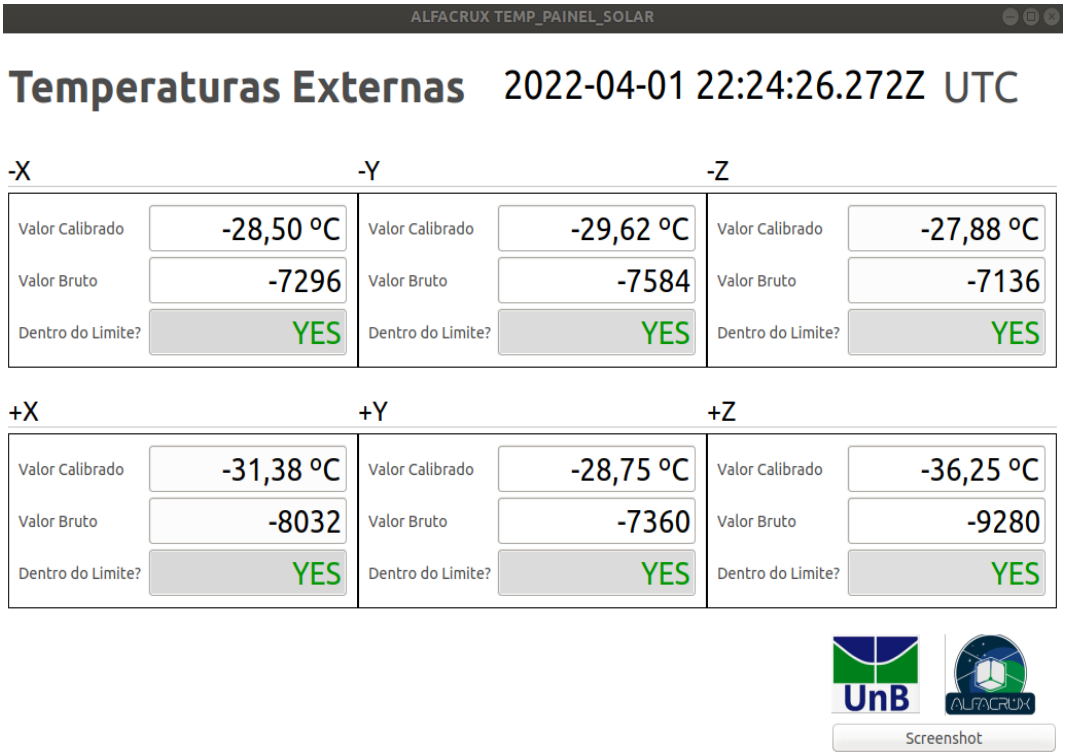


Figura 5.11 – Visualização das telemetrias do pacote TEMP_PAINEL_SOLAR.
Fonte: Autor

5.3 Envio de Telecomando

O primeiro passo para fazer o envio de telecomando é rodar o Cosmos com o comando "*ruby cosmos/tools/Launcher*"e abrir o *Command and Telemetry Server*. Em seguida, é preciso compilar e rodar o simulado do AlfaCrux, os comandos utilizados no terminal para essas ações são: "*make*"e "*./alfacrux.exe*".

Ao rodar o simulador do AlfaCrux, o *socket* aguarda uma conexão ser estabelecida, como pode ser visto na Figura 5.12. A conexão é estabelecida apenas quando o botão *Connect* da Interface ALFACRUX for pressionado. Dessa forma, o terceiro passo é habilitar esse botão para a conexão ser estabelecida.

O campo *Connected?* no *Command and Telemetry Server* informa se a conexão foi estabelecida com sucesso ou não. Se nesse campo estiver com a mensagem "*true*"em verde, a conexão foi estabelecida com sucesso, além disso, no terminal do simulador AlfaCrux é informado que o simulador foi conectado com o cliente (Fig: 5.13). Se a mensagem for "*attempting* em vermelho, a conexão não foi estabelecida e o simulador informa uma mensagem de erro ou fica aguardando uma conexão, dependendo do problema ocorrido.

Dois principais erros de conexão que podem ocorrer são: 1) O controlador habilitar o botão *Connect* antes de o simulador AlfaCrux ser inciado e 2) A conexão entre o Cosmos e o simulador ser fechada de forma incorreta e o simulador continuar com o endereço ocupado, não possibilitando a comunicação.

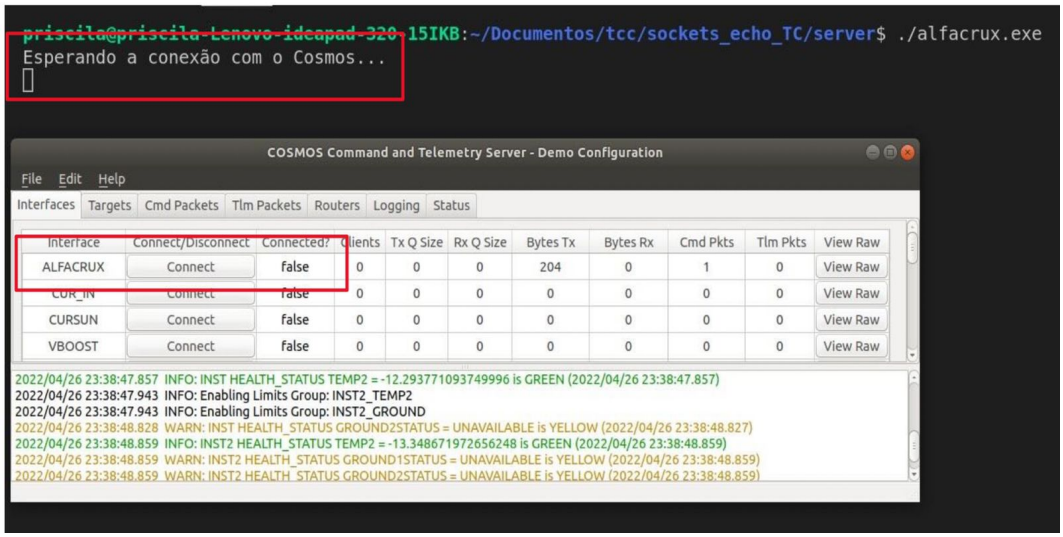


Figura 5.12 – Simulador AlfaCrux aguardando uma conexão ser estabelecida.
Fonte: Autor

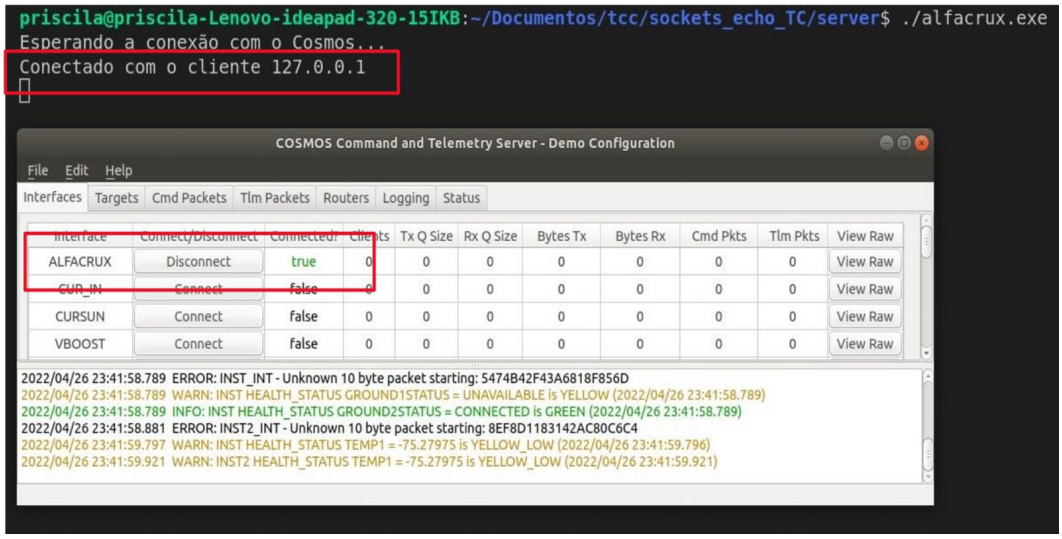
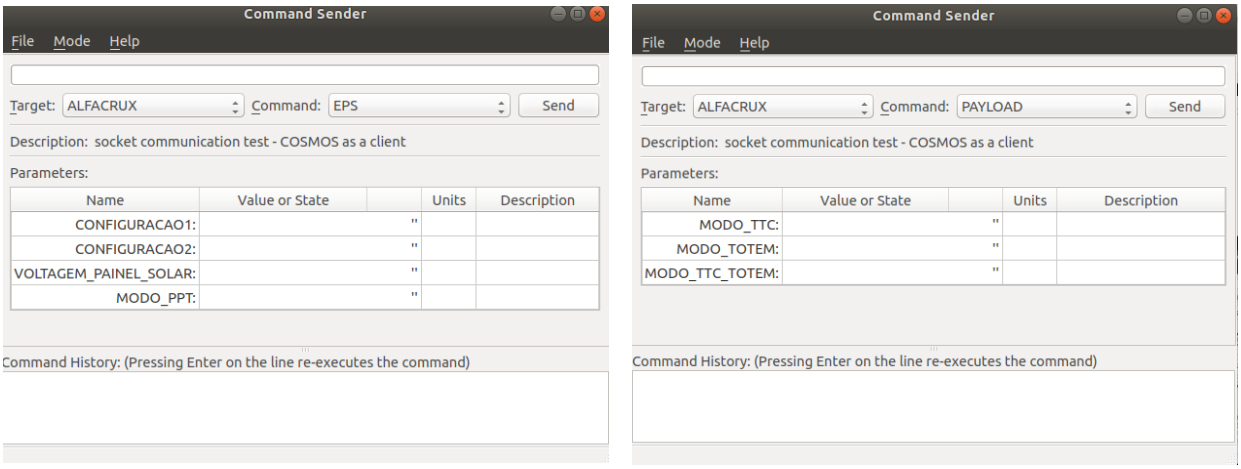


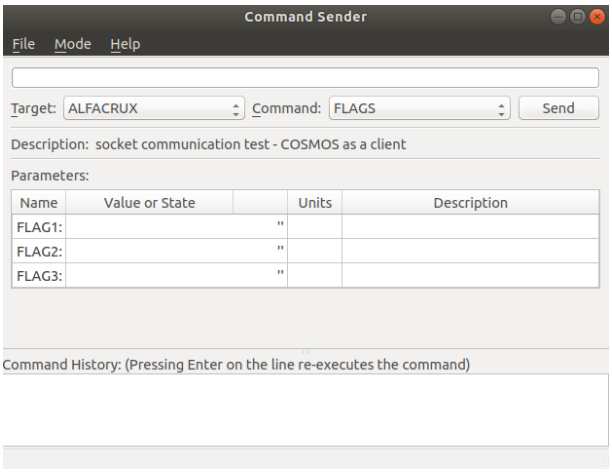
Figura 5.13 – Conexão estabelecida entre o simulador AlfaCrux e o Cosmos.
Fonte: Autor Fonte: Autor

Após estabelecida a comunicação entre o Cosmos e o simulador, é possível inciar o envio dos telecomandos. Os comandos são enviados pelo aplicativo *Command Sender*. O *Target* para simular o envio de telecomandos precisa ser o ALFACRUX, o qual tem 3 tipos de comando. Cada comando tem seus parâmetros, que podem ser visualizados na Fig. 5.14.

A Figura 5.15 mostra o envio de comandos do EPS. O valor enviado de cada parâmetro foi diferente para que o valor recebido pelo simulador possa ser validado. No campo *Value or Status* pode ser escrito qualquer comando que esteja dentro do número de *bits* definidos no código do Cosmos, a informação enviada depende da rotina de operação. O envio do telecomando é feito apenas após apertar o botão *Send*.



(a) (b)



(c)

Figura 5.14 – Parâmetros de cada tipo de comando, a) EPS, b) PAYLOAD e c) FLAGS.
Fonte: Autor

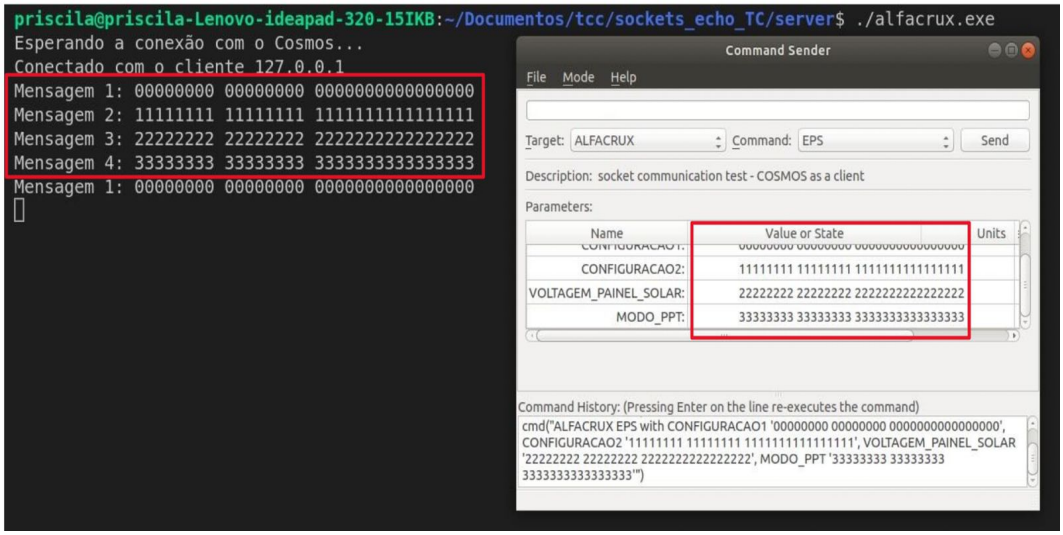


Figura 5.15 – Envio de comando do Cosmos para o simulador AlfaCrux.
Fonte: Autor

5.4 Tempo de revisita da missão da AlfaCrux

O tempo de revisita é um importante parâmetro para missões satelitais, sobretudo aquelas de imageamento e monitoramento climático. Este parâmetro diz respeito a quantas passagens sobre a cobertura da antena da estação terrena o satélite fará durante por dia e qual período ocorrerá a passagem. Assim, é possível fazer fazer as coletas de dados captados pelo satélite, bem como enviar comandos.

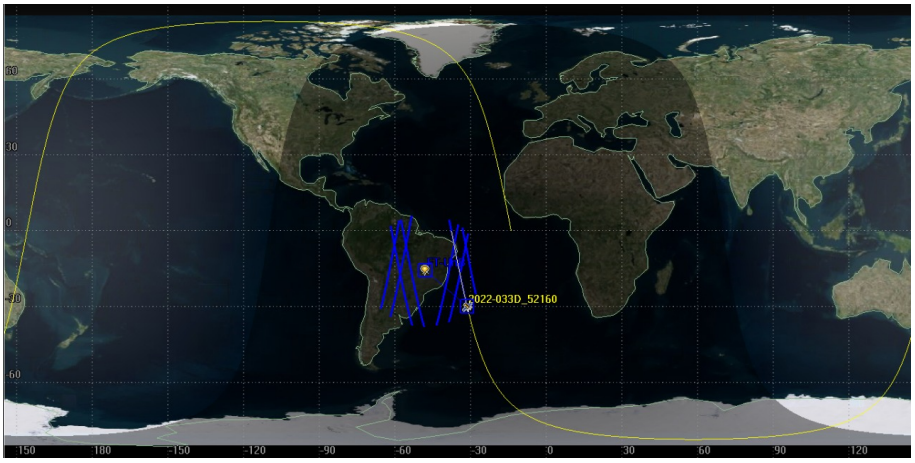
A missão AlfaCrux, devido aos seus parâmetros orbitais, é previsto entre 2 a 4 passagens diárias sobre a cobertura da estação solo. Deste modo, a fim de prever quantas passagens diárias e qual a sua duração em uma determinada data, foi realizada uma simulação no software Systems Tool Kit (STK) - AGI. A Tabela 5.4 apresenta o tempo de revisita do AlfaCrux entre o período de 12 de abril a 13 de abril de 2022, embora são apresentados os dados de revisita para dois dias consecutivos é possível prever as passagens para o tempo de vida útil da missão.

Tabela 5.4 – Tempo de revisita do AlfaCrux entre 12 de abril a 13 de abril de 2022.

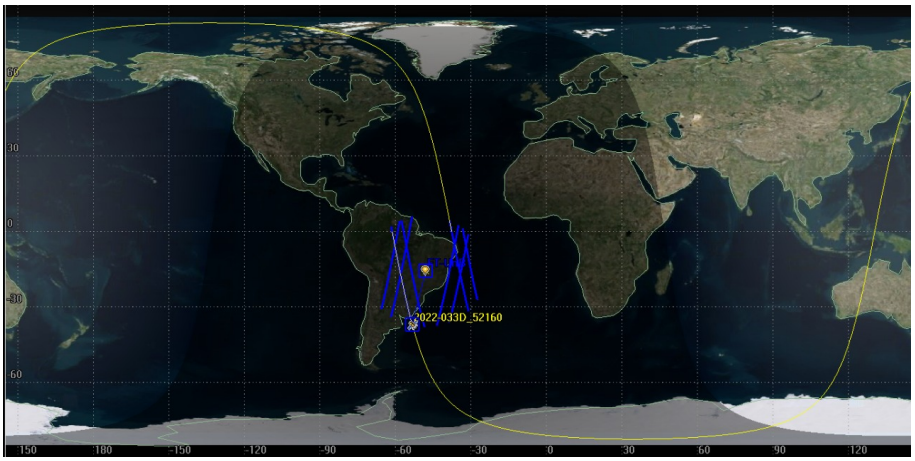
Passagem	**Data**	**Horário UTCG**	**Duração [s]**
1	12/04	01:19:37.984	573.214
2	12/04	02:52:51.210	626.063
3	12/04	13:18:15.185	633.784
4	12/04	14:52:35.120	555.436
5	13/04	01:02:27.251	448.009
6	13/04	02:34:20.804	673.367
7	13/04	13:00:47.612	557.508
8	13/04	14:33:42.599	634.141

Fonte: Autor

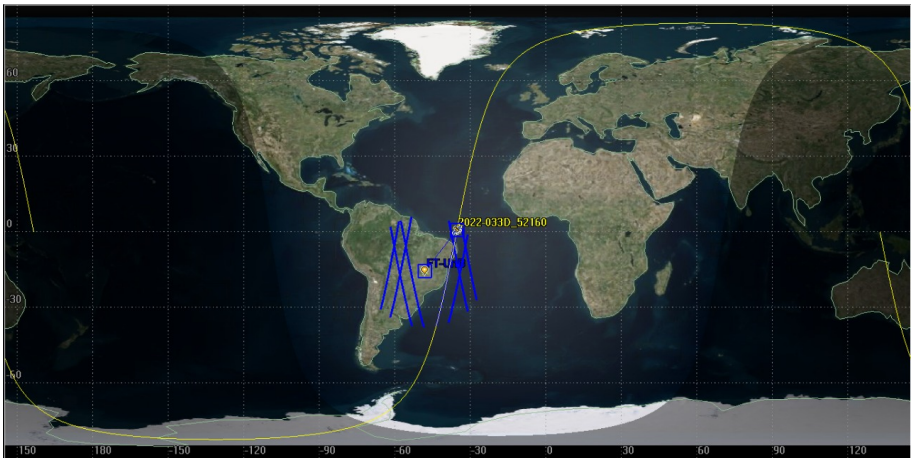
Através da simulação é possível prever a região de passagem do satélite. A Figura 5.16 apresenta as três primeiras passagens no dia 12 de abril de 2022, em que os segmentos de retas azuis indicam a região de passagem sob cobertura da antena da estação terrena. Deste modo, é possível sempre ter uma equipe disponível na estação solo de modo a monitorar o satélite durante o período de passagem, bem como fazer envio de telecomandos quando necessário.



(a)



(b)



(c)

Figura 5.16 – Passagens do AlfaCrux no dia 12/04/2022 na estação solo da UnB, a) primeira passagem às 01:19:37.984, b) segunda passagem às 02:52:51.210 e c) terceira passagem às 13:18:15.185.

Fonte: Autor

Outra forma de visualizar os dados de passagem é através do gráfico de revisita. A Figura 5.17 apresenta uma clássica representação do tempo de revisita por hora da passagem. As barras

vermelhas verticais indicam a duração da passagem do satélite em dado período do dia. Os dados apresentados são inerentes a revisita do AlfaCrux no período exposto anteriormente nesta seção.

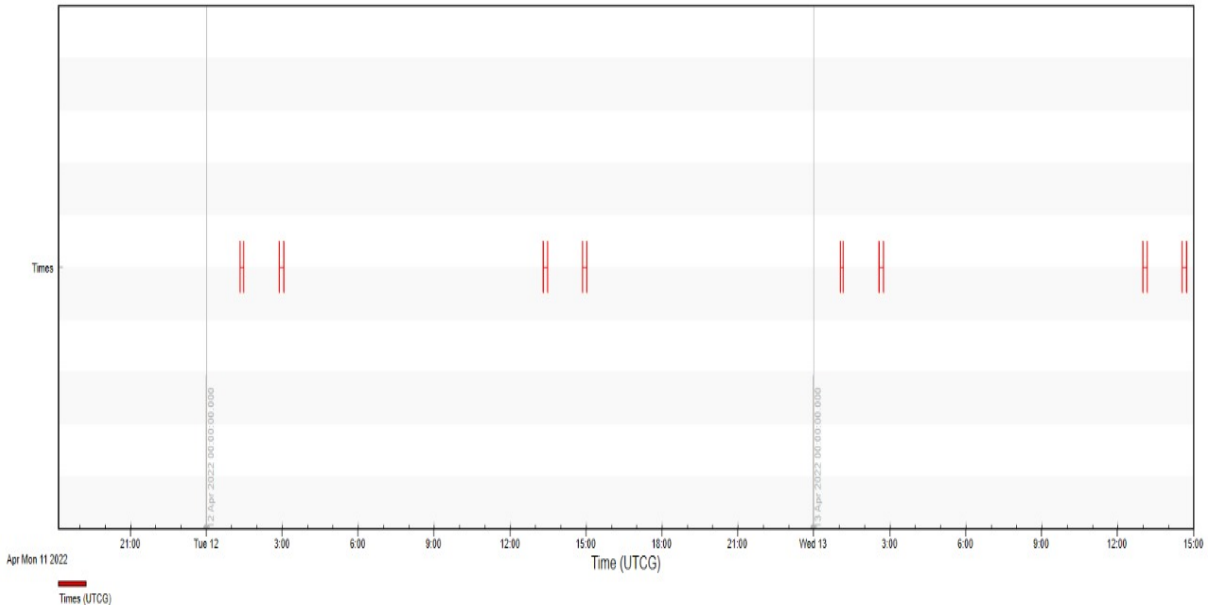


Figura 5.17 – Gráfico de revisita do AlfaCrux na estação solo da UnB.
Fonte: Autor

6 CONSIDERAÇÕES FINAIS

6.1 Conclusões

O objetivo final desse trabalho constitui no desenvolvimento de um sistema para a visualização de Telemetrias para o *CubeSat* AlfaCrux com dados reais da missão e a simulação de envio de telecomando para o AlfaCrux, a partir da *framework* Cosmos. O presente trabalho, também, tem o objetivo de inserir o conhecimento sobre o Cosmos na Universidade para que possa ser utilizado pelos alunos para missões futuras.

Para isso, foi criado telas exemplos com o recebimento de dados de um *software* simulado por conexão TCP/IP. Os dados enviados do *software* eram controlados e conhecidos com o objetivo de validar a conexão e o envio e o recebimento dos dados. Além disso, foi criado exemplos de telas para apresentar as possibilidades da ferramenta utilizada.

A conexão entre a ferramenta Cosmos e o *software* simulador, o envio e o recebimento das telemetrias simuladas se mostraram eficientes para atender os objetivos do trabalho. A conexão foi feita corretamente e os dados recebidos foram os mesmos que os dados enviados, sem nenhuma interferência ou ruído.

As telas de telemetrias, também, satisfazem os objetivos do presente trabalho, mostrando que é possível a visualização dos dados de forma personalizada e organizada. Bem como, a visualização dos limites das telemetrias, o tempo em que a telemetria foi recebida e conversão dos dados recebidos. Foi destacado que a forma que os dados são apresentados influenciam na visualização deles, o que pode tornar o entendimento dos dados mais intuitivo ou não.

Validado a comunicação e o recebimento das telemetrias, outra parte do desenvolvimento do presente trabalho foi enviar dados reais do AlfaCrux para o Cosmos e apresentar essas informações de forma clara e objetiva. O recebimento das telemetrias do AlfaCrux é feita por meio de planilhas na extensão csv, o que cumpre o objetivo de realizar testes com dados reais de uma missão.

No entanto, é possível implementar melhorias no processo de adquirir essas informações, por exemplo, ter acesso acesso direto ao banco de dados da missão. Esse procedimento não foi realizado nesse trabalho pois a missão ainda está em um período inicial, no qual, muitos processos ainda estão sendo compreendidos, implementados e depurados.

O objetivo relacionado ao telecomando foi cumprido no trabalho. É implementando uma simulação na qual recebe telecomandos do Cosmos, no entanto, esses telecomandos são criados de forma genérica. Esse conteúdo foi abordado mais superficialmente, podendo ser melhor explorado em trabalhos futuros, com a missão AlfaCrux operando em modo nominal.

Por fim, esse trabalho descreve as ferramentas utilizadas no Cosmos, mostra de forma detalhada como é criado uma interface tanto para o recebimento de telemetria como para o envio de telecomando, apresenta exemplos de telas e como esses exemplos são construídos e mostra como realizar algumas operações dentro dos aplicativos do Cosmos, cumprindo o objetivo de ser um documento base para que alunos da universidade utilizarem futuramente com o objetivo de desenvolver a interface de controle e comando de missões futuras.

Deste modo, conclui-se que este trabalho atingiu os objetivos e resultados esperados. Embora possa haver melhorias a serem implementadas, os resultados obtidos são válidos, aplicados para a missão AlfaCrux, flexíveis para ser utilizados em outras missões de *CubeSat* real e agregam conhecimento para trabalhos futuros dentro da Universidade de Brasília.

6.2 Trabalhos futuros

Com o objetivo de adquirir maior conhecimento da ferramenta Cosmos e futuramente ser possível implementar uma interface completa de controle e comando para missões de *CubeSat* dentro da universidade, é proposto algumas melhorias:

- Realizar a comunicação do Cosmos diretamente com o banco de dados do AlfaCrux para o recebimento de telemetrias;
- Utilizar *scripts* para realizar simulações de envio de telecomandos;
- Implementar uma conexão do Cosmos com interface *web*;
- Validar completamente a operação do AlfaCrux utilizando a solução baseada no Cosmos;
- Conectar o Cosmos com telemetrias recebidas por radioamador.

Referências

ALEN-SPACE. Alén space joins the development of a nanosatellite for the alfacrux mission from the university of brasilia. Acesso em: 28, out, 2021, 2021. Disponível em: <https://alen.space/alen-space-develops-a-nanosatellite-for-the-alfa-crux-mission-from-the-university-of-brasilia/>. Citado na página 47.

ALEN-SPACE. Products for small satellites. Acesso em: 28, out, 2021, 2021. Disponível em: <https://alen.space/products-for-small-satellites/>. Citado na página 51.

ALEN-SPACE. *TOTEM Motherboard Datasheet*. [S.l.], 2021. Citado na página 49.

ARGOS. Argos user's manual. Acesso em: 18, out, 2021, 2016. Disponível em: <https://www.argos-system.org/manual/index.html#3-location/32_principle.htm>. Citado na página 38.

BALLAEROSPACE. Directory structure. Acesso em: 22, out, 2021, 2021. Disponível em: <https://cosmosc2.com/docs/v4/structure>. Citado 3 vezes nas páginas 39, 43 e 57.

BALLAEROSPACE. Github. Acesso em: 20, out, 2021, 2021. Disponível em: <https://github.com/BallAerospace/COSMOS>. Citado na página 39.

BALLAEROSPACE. Installation. Acesso em: 20, out, 2021, 2021. Disponível em: <https://cosmosc2.com/docs/v4/installation>. Citado na página 39.

BALLAEROSPACE. Interface configuration. Acesso em: 22, out, 2021, 2021. Disponível em: <https://cosmosc2.com/docs/v5/interfaces>. Citado na página 60.

BALLAEROSPACE. Requerements and design. Acesso em: 04, out, 2021, 2021. Disponível em: <https://cosmosc2.com/docs/requirements>. Citado 5 vezes nas páginas 39, 40, 41, 42 e 44.

BALLAEROSPACE. Telemetry. Acesso em: 23, out, 2021, 2021. Disponível em: <https://cosmosc2.com/docs/v4/telemetry>. Citado na página 59.

BALLAEROSPACE. Telemetry screens. Acesso em: 22, out, 2021, 2021. Disponível em: <https://cosmosc2.com/docs/v4/screens>. Citado 2 vezes nas páginas 41 e 42.

BARY, B. R.; RAYMOND, L. J.; KENNETH, P. M. Telemetry, tracking and control for satellite cellular communication systems. Acesso em: 19, out, 2021, 1993. Disponível em: <https://patents.google.com/patent/US5187805A/en>. Citado na página 33.

BEECH, W. A.; NIELSEN, D. E.; TAYLOR, J. Ax.25 link access protocol for amateur pakcet radio. TAPR, n. Version 2.2, 1998. Citado na página 30.

BRUNO, P. Z.; ROMARIO, T. Geotecnologias: Discussões e análies a respeito da evolução dos sistemas global de navegação por satélites - gnss. In: *Revista Eletrônica em Gestão, Educação e Tecnologia Ambiental*. Rio Grande do Sul, Brasil: [s.n.], 2015. Citado na página 21.

CANADA, G. of. Satellites in our everyday lives. Acesso em: 17, out, 2021, 2020. Disponível em: <https://www.asc-csa.gc.ca/eng/satellites/everyday-lives/default.asp>. Citado na página 16.

CCSDS. Overview of space communications protocols. National Aeronautics an Space Administration, n. Issue 3, 2014. Citado na página 32.

CGEE, C. de Gestão e E. E. *Resumo Executivo - CubeSats*. [S.l.]: CGEE, 2018. Citado 5 vezes nas páginas 16, 17, 23, 25 e 26.

CONCERNED-SCIENTISTS, U. of. Ucs satellite database. in-depth details on the 4,084 satellites currently orbiting earth, including their country of origin, purpose, and other operational details. Acesso em: 18, out, 2021, 2021. Disponível em: <https://www.ucsusa.org/resources/satellite-database>. Citado na página 25.

CONFAP. Projeto alfacrux, apoiado pela fapdf, começa a desenvolver estação de solo que vai fazer rastreamento e controle do nanossatélite. Acesso em: 21, out, 2021, 2020. Disponível em: <https://confap.org.br/news/projeto-alfa-crux-apoiado-pela-fapdf-comeca-a-desenvolver-estacao-de-solo-que-\vai-fazer-rastreamento-e-controle-do-nanossatelite/>. Citado 2 vezes nas páginas 46 e 47.

CORPORATION, R. S. S. Chronicle of soviet-russian space program. Acesso em: 17, out, 2021, 2016. Disponível em: <https://www.thepolicychronicle.co.in/chronicle-of-soviet-russian-space-program/>. Citado 2 vezes nas páginas 16 e 20.

COUNCIL, N. R. *Technology for Small Spacecraft*. Washington, DC: The National Academies Press, 1994. ISBN 978-0-309-05075-3. Disponível em: <https://www.nap.edu/catalog/2351/technology-for-small-spacecraft>. Citado 2 vezes nas páginas 20 e 26.

DELMONDES, R. da C. *Proposta de uma Metodologia para o Desenvolvimento de um Subsistema de Telemetria e Comando para Plataformas Estratosféricas*. Dissertação (Trabalho de Conclusão de Curso) — Universidade de Brasília, 2019. Citado 2 vezes nas páginas 16 e 32.

DUBOS, G.; CASTET, J.-F.; SALEH, J. Statistical reliability analysis of satellites by mass category: Does spacecraft size matter? *Acta Astronautica*, v. 67, 09 2010. Citado na página 23.

ECSS. Space engineering. In: *European Coopertarion for Space Standardization*. Noordwijk, Holanda: [s.n.], 2000. Citado 2 vezes nas páginas 27 e 28.

FAPDF. Alfacrux. Acesso em: 21, out, 2021, 2020. Disponível em: <https://www.fap.df.gov.br/alfa-crux/>. Citado na página 46.

FONSECA, H. N. *Estudo de viabilidade e proposta de topologia de um sistema de gerenciamento de energia para nanosatélites da constelação AlfaCrux*. Dissertação (Trabalho de Conclusão de Curso) — Universidade de Brasília, 2020. Citado 2 vezes nas páginas 46 e 47.

FORTESCUE, P.; STARK, J.; SWONERD, G. *Spacecraft systems engineering*. [S.l.]: Wiley, 2003. Citado na página 28.

GOMSPACE. *NanoCom AX100 Datasheet Long-range software configurable VHF/UHF transceiver*. [S.l.], 2016. Citado na página 51.

GUEST, A. N. *Handbook of Satellite Applications: Telemetry, Tracking and Command (TT&C)*. [S.l.]: Springer, 2016. Citado 6 vezes nas páginas 32, 33, 34, 35, 36 e 37.

HARTWELL, K.; LIGHTSEY, G. Design and creation of mission control center for georgia tech satellites. In: *Space Systems Design Lab (SSDL)*. Atlanta, Estados Unidos: [s.n.], 2021. Citado 5 vezes nas páginas 16, 17, 23, 27 e 38.

HURRICANES. Low earth orbit satellites. Acesso em: 17, out, 2021, 2020. Disponível em: <http://www.hurricanescience.org/science/observation/satellites/poes/>. Citado na página 21.

JORDI, P. suari; CLARK, T.; WILLIAM, A. Development of the standard cubesat deployer and a cubesat class picosatellite. In: *IEEE Aerospace Conference Proceedings*. Montana, Estados Unidos: [s.n.], 2001. Citado 2 vezes nas páginas 25 e 26.

KEEESEE, C. E. Satellite telemetry, tracking and control subsystems. Acesso em: 18, out, 2021, 2003. Disponível em: <https://ocw.mit.edu/courses/aeronautics-and-astronautics/16-851-satellite-engineering-fall-2003/lecture-notes/l20_satellitettc.pdf>. Citado na página 36.

LABRE-RS. Lista de faixas e subfaixas do serviço radioamador no brasil. Acesso em: 30, out, 2021, 2021. Disponível em: <https://labre-rs.org.br/plano-de-faixas/>. Citado na página 35.

LEFFKE, Z. Virginia tech ground station tnc interfacing tutorial. In: *Virginia Tech*. Virgínia, Estados Unidos: [s.n.], 2018. Citado 2 vezes nas páginas 30 e 31.

LEY, W.; WITTMANN, K.; HALLMANN, W. *Handbook of Space Technology*. [S.l.]: Wiley, 2009. Citado 6 vezes nas páginas 20, 22, 27, 28, 29 e 38.

MCTI. Projeto alfacrux começa a desenvolver estação de solo que vai fazer rastreamento e controle do nanossatélite. Acesso em: 29, set, 2021, 2020. Disponível em: <https://www.gov.br/aeb/pt-br/assuntos/noticias/projeto-alfa-crux-comeca-a-desenvolver-estacao-de-solo-\que-vai-fazer-rastreamento-e-controle-do-nanossatelite>. Citado 3 vezes nas páginas 17, 46 e 47.

MCTI. Aeb explica importância dos satélites e uso científico para a humanidade. Acesso em: 17, out, 2021, 2021. Disponível em: <https://www.gov.br/mcti/pt-br/acompanhe-o-mcti/noticias/2020/10/aeb-explica-importancia-dos-satelites-e-uso-cientifico-para-a-humanidade>. Citado na página 16.

MD, T. M. J. *Satellite Broadcasting Fundamentals*. [S.l.]: Lulu.com, 2019. Citado na página 21.

NASA. An early history of satellites timelines. Acesso em: 17, out, 2021, 2014. Disponível em: <https://www.jpl.nasa.gov/infographics/an-early-history-of-satellites-timeline>. Citado na página 20.

NASA. What is a satellite? Acesso em: 17, out, 2021, 2014. Disponível em: <https://www.nasa.gov/audience/forstudents/5-8/features/nasa-knows/what-is-a-satellite-58.html>. Citado na página 21.

NASA. What are smallsats and cubesats. Acesso em: 29, set, 2021, 2015. Disponível em: <https://www.nasa.gov/content/what-are-smallsats-and-cubesats>. Citado na página 17.

NASA. What is a satellite? Acesso em: 17, out, 2021, 2017. Disponível em: <https://www.nasa.gov/audience/forstudents/k-4/stories/nasa-knows/what-is-a-satellite-k4.html>. Citado na página 21.

NASA. Small spacecraft technology state of the art. Acesso em: 17, out, 2021, 2020. Disponível em: <https://www.nasa.gov/smallsat-institute/sst-soa>. Citado 3 vezes nas páginas 23, 24 e 25.

NASA. Tracking and data relay satellite (tdrs). Acesso em: 18, out, 2021, 2021. Disponível em: <https://www.nasa.gov/directorates/heo/scan/services/networks/tdrs_main>. Citado na página 34.

PELTON, J. N.; MADRY, S.; CAMACHO-LARA, S. *Handbook of Satellite Applications*. [S.l.]: Springer, 2016. Citado 3 vezes nas páginas 16, 21 e 22.

PELTON, J. N.; MADRY, S.; CAMACHO-LARA, S. *Handbook of Satellite Applications: Ground Systems for Satellite Application Systems for Navigation, Remote Sensing, and Meteorology*. [S.l.]: Springer, 2016. Citado na página 32.

PESQUISA-FAPESP. Pequenos ganham o espaço. nanossatélites são lançados em missões de coletas de dados que vão do monitoramento ambiental a testes de sistemas biológicos. Acesso em: 18, out, 2021, 2019. Disponível em: <https://revistapesquisa.fapesp.br/pequenos-ganham-o-espaco/>. Citado 2 vezes nas páginas 17 e 25.

PISACANE, V. L. *Fundamentals of Space Systems*. [S.l.]: Oxford, 2005. Citado na página 35.

POVO, C. do. Brasil terá nanossatélite para conectar áreas remotas do país. Acesso em: 21, out, 2021, 2020. Disponível em: <https://www.correiodopovo.com.br/jornalcomtecnologia/brasil-ter%C3%A1-nanossat%C3%A9lite-para-conectar-%C3%A1reas-remotas-do-pa%C3%ADs-1.495963>. Citado na página 47.

PROGRAM, C.-S. T. C. Cubesat design specification. 2020. Citado 2 vezes nas páginas 16 e 26.

PROGRAM, C.-S. T. C. Cubesat design specification. Rev 12. Citado na página 25.

REIS, L. R. et al. An onverview of the alfacrux cubesat mission for narrowband communication. In: $5^{ht} IAA Conference on University Satellite Missions and CubeSat Workshop. Roma, Itália : [s.n.], 2020. Citado 2 vezes nas páginas 48 e 50.$

RICHHARIA, M. *Satellite Communications Systems Design Principles*. [S.l.]: Macmillan, 1995. Citado na página 21.

RODRIGUES, J. E. O. *Processo de Referência para o desenvolvimento da arquitetura de uma estação terrena para pico e nanosatélites*. Dissertação (Mestrado) — Instituto de Pesquisas Espaciais, 2016. Citado 2 vezes nas páginas 23 e 27.

ROHLING, A. J. *A reference architecture for satellite systems operations*. Tese (Doutorado) — Instituto de Pesquisas Espaciais, 2018. Citado 2 vezes nas páginas 17 e 22.

SHIROMA, W. et al. Cubesats: A bright future for nanosatellites. *Open Engineering*, v. 1, 03 2011. Citado 2 vezes nas páginas 23 e 26.

SOARES, V. Satélite ajustado às regiões remotas. In: *Satélite ajustado às regiões remotas*. Distrito Federal, Brasil: [s.n.], 2020. Citado na página 47.

SURHONE, L. M.; TONNOE, M. T.; HENSSONOW, S. F. *Reed-Solomon Error Correction*. [S.l.]: Betascript Puclishing, 2010. Citado na página 52.

SWEETING, M. N.; UNDERWOOD, C. I. *Small Satellite Engineering and Applications*. John Wiley Sons, Ltd, 2011. 575-605 p. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119971009.ch18>. Citado 2 vezes nas páginas 23 e 24.

TOORIAN, A.; DIAZ, K.; LEE, S. The cubesat approach to space access. In: *2008 IEEE Aerospace Conference*. [S.l.: s.n.], 2008. p. 1–14. Citado na página 16.

VINHOTE, A. L. Satélite vai ajudar produtores rurais. Acesso em: 21, out, 2021, 2020. Disponível em: <https://www.agenciabrasilia.df.gov.br/2020/08/29/satelite-vai-ajudar-produtores-rurais/>. Citado na página 46.

WAYNE, S. A. et al. Cubesats: A bright future for nanosatellites. In: *Central European Journal of Engineering*. Hawaii, Estados Unidos: [s.n.], 2011. Citado 2 vezes nas páginas 24 e 25.

WERTZ, J. R.; LARSON, W. *Space Mission Analysis and Design*. [S.l.]: Springer Netherlands, 2005. Citado na página 25.

WERTZ, J. R.; LARSON, W. J. *Space Mission Analysis and Design*. [S.l.]: Microcosm, 1999. Citado na página 34.

WILLIAM, I. D. Architecture study of space-based satellite networks for nasa missions. In: *NASA*. Ohio, Estados Unidos: [s.n.], 2003. Citado na página 34.

APÊNDICE A – AlfaCrux

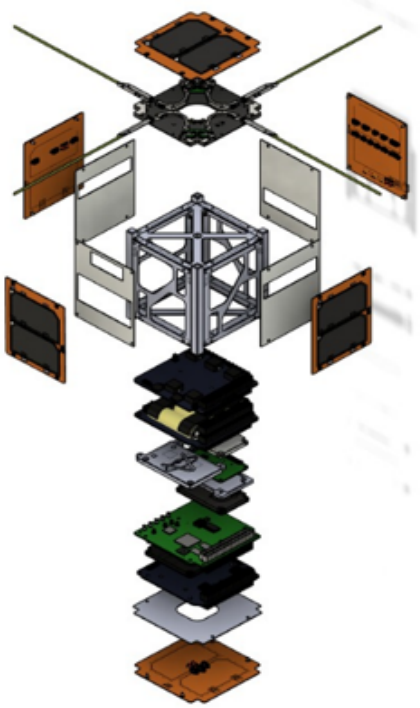


Figura A.1 – Vista explodida do AlfaCrux
Fonte: AlfaCrux (2022). Não publicado

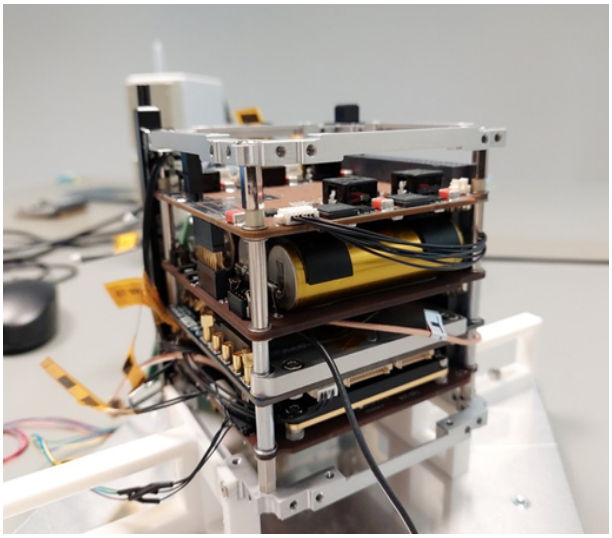


Figura A.2 – Parte interna do AlfaCrux
Fonte: AlfaCrux (2022). Não publicado

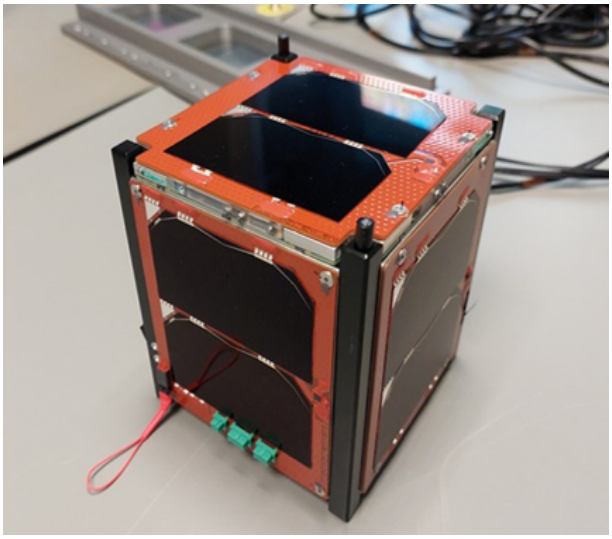


Figura A.3 – Parte externa do AlfaCrux
Fonte: AlfaCrux (2022). Não publicado

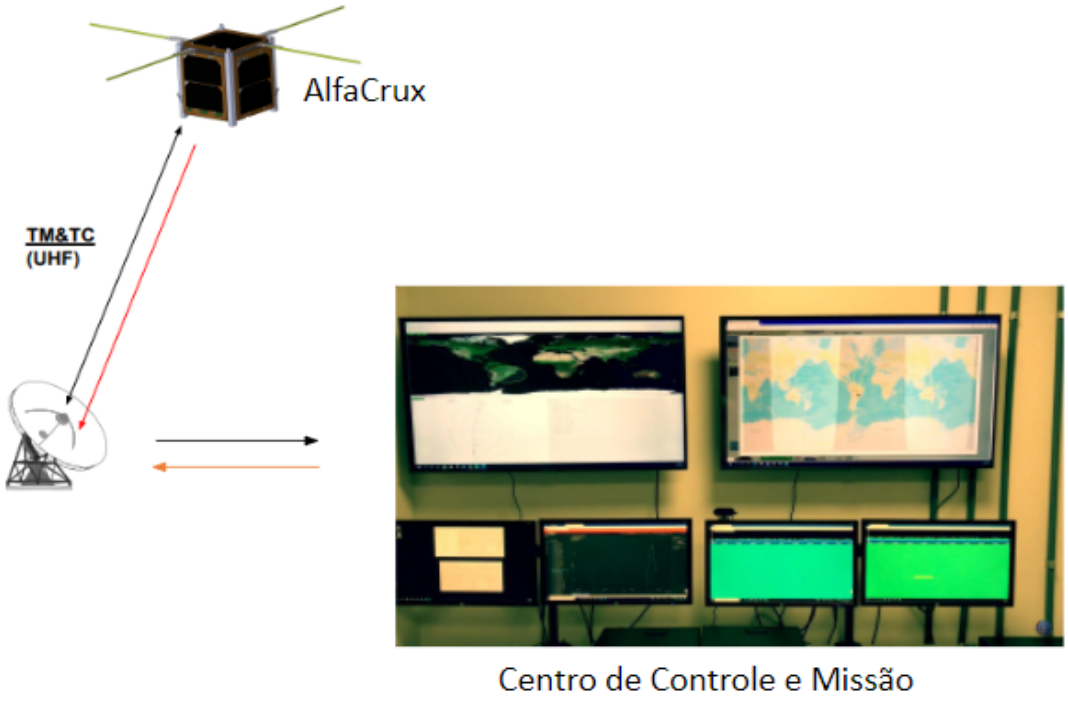


Figura A.4 – Interface segmento espacial - segmento solo do AlfaCrux
Fonte: Autor

(a)

(b)

(c)

Figura A.5 – Montagem da antena
Fonte: Autor