


**PROPOSTA DE UMA METODOLOGIA PARA O
DESENVOLVIMENTO DE UM SUBSISTEMA DE
TELEMETRIA E COMANDO PARA
PLATAFORMAS ESTRATOSFÉRICAS**



Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Mecânica

**PROPOSTA DE UMA METODOLOGIA PARA O
DESENVOLVIMENTO DE UM SUBSISTEMA DE
TELEMETRIA E COMANDO PARA
PLATAFORMAS ESTRATOSFÉRICAS**

Rômulo da Costa Delmondes

Orientadora: Andrea Cristina dos Santos

Dissertação de Mestrado em Sistemas Mecatrônicos

Publicação: ENM-DM ***/2019

Brasília-DF: 08/2019

Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Mecânica

**PROPOSTA DE UMA METODOLOGIA PARA O
DESENVOLVIMENTO DE UM SUBSISTEMA DE
TELEMETRIA E COMANDO PARA
PLATAFORMAS ESTRATOSFÉRICAS**

Rômulo da Costa Delmondes

**Dissertação submetida ao departamento de Engenharia Mecânica
da Faculdade de Tecnologia da Universidade de Brasília como
parte dos requisitos necessários para obtenção do grau de mestre
em Sistemas Mecatrônicos**

Aprovada por:

Andrea Cristina dos Santos, Prof.^a Dr.^a, UnB/PPMEC
(Orientadora)

Jones Yudi Mori Alves da Silva, Prof. Dr., UnB/PPMEC
(Examinador interno)

Renato Alves Borges, Prof. Dr., UnB/ENE
(Examinador externo)

Carlos Humberto Llanos Quintero, Prof. Dr., UnB/PPMEC
(Examinador suplente)

Brasília, 08/2019.

FICHA CATALOGRÁFICA

Rômulo da Costa Delmondes

PROPOSTA DE UMA METODOLOGIA PARA O DESENVOLVIMENTO DE UM SUBSISTEMA DE TELEMETRIA E COMANDO PARA PLATAFORMAS ESTRATOSFÉRICAS/ Rômulo da Costa Delmondes. – Brasil, 2019-264 p. : il. (algumas color.) ; 30 cm.

Orientador: Andrea Cristina dos Santos

Dissertação (mestrado) – Universidade de Brasília – UnB

Faculdade de Tecnologia

Programa de Pós-Graduação em Sistemas Mecatrônicos, 2019.

1. Model-Based Design 2. Modelo V 3. Telemetria 4. Comando I. Engenharia Mecânica. II. Universidade de Brasília. III. Faculdade de Tecnologia. IV. Desenvolvimento de um subsistema de telemetria e comando para plataformas estratosféricas.

REFERÊNCIA BIBLIOGRÁFICA Delmondes, C. R. (2019). DESENVOLVIMENTO DE UM SUBSISTEMA DE TELEMETRIA E COMANDO PARA PLATAFORMAS ESTRATOSFÉRICAS. Dissertação de Mestrado em Sistemas Mecatrônicos, Publicação ENM.DM-*** */2018, Departamento de Engenharia Mecânica, Universidade de Brasília, Brasília, DF, p.264.

CESSÃO DE DIREITOS NOME DO AUTOR: Rômulo da Costa Delmondes.

TÍTULO DA DISSERTAÇÃO DE MESTRADO: DESENVOLVIMENTO DE UM SUBSISTEMA DE TELEMETRIA E COMANDO PARA PLATAFORMAS ESTRATOSFÉRICAS. **GRAU/ANO:** Mestre/2019.

É concedida à Universidade de Brasília permissão para reproduzir cópias desta dissertação de mestrado e para emprestar tais cópias somente para propósitos acadêmicos e científicos. O autor reserva outros direitos de publicação e nenhuma parte desta dissertação de mestrado pode ser reproduzida sem a autorização por escrito do autor.

Rômulo da Costa Delmondes

Rua 19B Qd.40 Lt.18 Cs.01, Garavelo B - Goiânia

74354-115, Goiânia, GO, Brasil

romulodelmondes@gmail.com

Dedico este trabalho aos meus pais, Raimundo e Rozineide, guardiões incansáveis de todos os meus passos, responsáveis pela entrega das ferramentas certas, nos momentos certos.

Agradecimentos

Aos meus pais, exemplos de amor, perseverança, foco, força e fé, espelhos do meu exercício diário de evolução, através dos caminhos do conhecimento, fraternidade, honestidade e respeito. À minha irmã, modelo de garra, otimismo e dinamismo, pessoa inquieta e contagiante, em constante procura pela felicidade e paz espiritual. Sigo sempre em frente calcado nos exemplos ensinados e nos modelos copiados, observados na convivência diária ao lado de vocês.

Meu respeito e admiração ao meu tio e padrinho Ubirajara Avelino e, minha tia e madrinha Eumélia Avelino, pessoas que desde o princípio desempenham papéis de imensa relevância em cada tijolo que compõe a minha construção.

À Prof.^a Dr.^a Andrea Santos, que me recebeu como orientando, tornando o processo do mestrado mais leve e humano, pela sensibilidade em entender as dificuldades que imperaram durante esta caminhada. Sou muito grato pelos ensinamentos, na compreensão e descoberta do mundo acadêmico e, por ser esta, referência em termos de amor à profissão e empenho na conexão entre a academia e mercado profissional.

Ao Prof. Dr. Renato Borges, meus agradecimentos pela paciência e confiança depositada em participar do projeto LAICAnSat sob sua coordenação. Fazer parte deste grupo e poder agregar com a minha contribuição neste estudo foi uma honra, dada a grande importância, magnitude e complexidade no âmbito científico. Nosso convívio e trabalho em equipe me concederam importantes experiências e interesses profissionais, em virtude dos desafios impostos.

Aos membros do Ministério de Ciência, Tecnologia, Informação e Comunicações, Dr. em Radiodifusão Flávio Lima e o Ms. em Sistemas Mecatrônicos Gilvanson Cavalcante, pelas orientações que nortearam o desenvolvimento desta pesquisa no âmbito das Telecomunicações.

Aos professores João Batista e Kelias de Oliveira. Doutores que me acompanham desde os primeiros passos nos idos da década de 90, no curso técnico em Eletrônica da saudosa ETFG, e que tive a satisfação de reencontrá-los na especialização em Telecomunicações - Prédios Inteligentes do IFG, colaboradores obstinados no apoio desta pesquisa, abrindo as portas dos laboratórios e disponibilizando tempo para realização de testes e troca de experiências.

Aos primos consanguíneos. Hitalo Delmondes e Fabiano Avelino pessoas formidáveis, aliados no processo de luta e crescimento diário. Aos grandes amigos. Ivan Martins, João Pedro, João Vitor Cantarelli pelos incontestáveis apoios técnicos. Paulo Sérgio pelos

conselhos nas noites de boemia. Pablo Diniz pela grande contribuição acadêmica e científica e, no acolhimento em sua residência nas temporadas de viagens a Brasília, momentos em que pudemos colocar em dia as conversas e lembranças, em virtude do grande intervalo afastados por ocasião da profissão. Silvio Oliveira amigo desde a graduação, companheiro de luta na Associação Brasileira de Engenheiros Eletricistas - Seção Goiás e grande incentivador deste trabalho, suas contribuições foram decisivas, no âmbito geral desta pesquisa. Kassio Bueno, Marcos Machado, Marcus Hermínio, Marianne Bortoletto, Peterson Caparrosa, Rafael Gramont e Salustiano Faria, grandes incentivadores e motivadores. Ao casal Rodrigo Rodrigues e Ana Rafaela, ambos conhecedores dos desafios impostos pela UnB, ela pelo pronto-atendimento nos momentos de auxílio acadêmico, ele pela longa amizade, um irmão que ganhei nos tempos de ETEG. Aos amigos de UnB, Marlete e Everaldo, juntos nesta jornada, cada um envolvido com sua pesquisa específica, mas sempre presentes, ouvindo e apoiando. As amigas Lorena Gonzaga e Melissa Vieira, pelos conhecimentos em regência, concordância e gramática, contribuições enriquecedoras para o amadurecimento desta dissertação.

Resumo

O trabalho em análise teve como objeto de estudo o desenvolvimento de um subsistema de telemetria (*downlink*) e comando (*uplink*). O objetivo principal consistiu em modelar, através da metodologia de desenvolvimento de sistemas, um subsistema de comunicação ponto-a-ponto, do tipo *half-duplex*, dedicado à missões espaciais aplicado em plataformas estratosféricas, atendendo ao padrão CubeSat. Através de uma metodologia híbrida, baseada na modelagem *Model-Based Design* (MBD), que agrega conceitos das engenharias de produtos e serviços, este subsistema permite um enfoque colaborativo de multidisciplinaridade, integrado ao Modelo V que permeia o desenvolvimento de sistemas e seus domínios específicos, auxiliando no processo para concepção do protótipo final. Em um primeiro momento, foram realizadas as modelagens dos domínios específicos - eletrônica, mecânica e *software*, mapeando os requisitos e definindo as arquiteturas físicas e lógicas do sistema, levando à construção do modelo virtual, o qual permitiu uma análise, em simulação, do comportamento do sistema, por meio dos métodos de testes do MBD: MIL (*Model-In-The-Loop*), SIL (*Software-In-The-Loop*) e HIL (*Hardware-In-The-Loop*), os quais possibilitaram iterações em *loop* para o aprimoramento em tempo real do sistema dinâmico. Por fim, o estudo alcançou a elaboração do protótipo lançando mão de equipamentos de baixo custo e disponíveis no mercado (COTS, em inglês *comercial off-the-shelf*). Concomitante à etapa de construção, foram ajustados os algoritmos de transmissão, recepção e demais funções auxiliares, refinados a partir do código gerado pelo modelo virtual, cuja modelagem antecipada garantiu a correta operação das funcionalidades presentes nas estações terrestre (ET) e embarcada (EE), que compõem o sistema.

Palavras-chave: model-based design. modelo V. telemetria. comando.

Abstract

The work under analysis had as object of study the development of a telemetry subsystem (downlink) and command (uplink). The main objective was to model, through the systems development methodology, a half-duplex point-to-point communication subsystem dedicated to space missions applied to stratospheric platforms, meeting the CubeSat standard. Through a hybrid methodology based on Model-Based Design (MBD) modeling that brings together concepts from product and service engineering, this subsystem allows a collaborative approach to multidisciplinary, integrated with Model V that permeates systems development and its specific domains. assisting in the process for designing the final prototype. At first, the modeling of the specific domains - electronic, mechanical and software, was performed, mapping the requirements and defining the physical and logical architectures of the system, leading to the construction of the virtual model, which allowed a simulation analysis of the system behavior. Through MBD testing methods: Model-In-The-Loop (MIL), Software-In-The-Loop (SIL) and Hardware-In-The-Loop (HIL) testing, allowing loop iterations for enhancement real-time dynamic system. Finally, the study reached the elaboration of the prototype, making use of low cost and commercially available equipment (COTS), concomitant with the construction stage, the transmission, reception algorithms and other auxiliary functions were refined from the code generated by the virtual model, whose early modeling ensured the correct operation of the functionalities present in the ground (ET) and embedded (EE) stations that make up the system.

Key-words: model-based design. model V. telemetry. command.

Lista de ilustrações

Figura 1 – Categorias dos pequenos satélites.	7
Figura 2 – Configurações para CubeSats.	9
Figura 3 – Configurações satélite <i>versus</i> massa.	10
Figura 4 – Subsistemas dos satélites.	11
Figura 5 – Efeito Doppler - Rastreo de satélites.	19
Figura 6 – Sistema de comunicação ponto-a-ponto.	22
Figura 7 – Classificação dos sistemas de transmissão.	22
Figura 8 – Zona de Fresnel.	28
Figura 9 – O espectro eletromagnético e suas aplicações na comunicação.	29
Figura 10 – Regiões de Administração do Espectro de RF.	30
Figura 11 – Ponto de vista do modelo do produto.	39
Figura 12 – Técnica hierárquica de modelagem OO.	40
Figura 13 – Abordagem <i>Top-down</i>	41
Figura 14 – Tipos de <i>ports</i>	42
Figura 15 – Módulo unificado.	43
Figura 16 – Modelo V para desenvolvimento de sistemas.	44
Figura 17 – Fluxo principal de atividade de projeto mecânico, eletrônico e de <i>software</i>	45
Figura 18 – Matriz de Adoção.	47
Figura 19 – Matriz de Adoção integrada ao Modelo V.	50
Figura 20 – MBD simplificado.	51
Figura 21 – Aplicação Modelo V.	53
Figura 22 – Sequência de técnicas para verificação de modelos	54
Figura 23 – Gráfico de Pareto.	63
Figura 24 – Diagrama de blocos geral.	91
Figura 25 – Domínio específico no diagrama de blocos.	92
Figura 26 – Mapa de fabricantes - Domínio eletrônica.	93
Figura 27 – Mapa de fabricantes - Domínio mecânica.	100
Figura 28 – Padrão de radiação do sinal - Antena omnidirecional.	101
Figura 29 – Padrão de radiação do sinal - Antena direcional.	101
Figura 30 – Diagrama de caso de uso.	106
Figura 31 – Diagrama de atividades.	107
Figura 32 – Modelo de arquitetura lógica.	109
Figura 33 – Diagrama de contexto da arquitetura para o subsistema de comunicação.	109
Figura 34 – Diagrama de fluxo da arquitetura para o subsistema de comunicação.	111
Figura 35 – Arquitetura - Domínio Físico.	113
Figura 36 – Arquitetura integrada.	114

Figura 37 – Sequência de testes.	116
Figura 38 – Integração Modelo Virtual.	117
Figura 39 – Detalhamento do bloco gerador de pulso.	119
Figura 40 – Detalhamento do bloco radiotransmissor Tx.	120
Figura 41 – Detalhamento do bloco antena Tx e Rx.	121
Figura 42 – Detalhamento do bloco espaço livre.	122
Figura 43 – Detalhamento do bloco alvo.	124
Figura 44 – Detalhamento do bloco radiotransmissor Rx.	124
Figura 45 – Modelo Virtual simplificado.	125
Figura 46 – Modelo Virtual completo.	127
Figura 47 – Gráfico do gerador de pulso.	129
Figura 48 – Gráfico do analisador de frequência.	130
Figura 49 – Sinal completo na recepção.	132
Figura 50 – Sinal proveniente de ruídos na transmissão.	133
Figura 51 – Sinal de interesse na recepção	134
Figura 52 – Fluxograma teste SIL.	136
Figura 53 – Integração componentes EE - Plataforma.	137
Figura 54 – Diagrama físico EE.	138
Figura 55 – Diagrama de integração lógica EE.	140
Figura 56 – Integração componentes ET - Plataforma.	141
Figura 57 – Diagrama físico ET.	142
Figura 58 – Diagrama da Unidade de Condicionamento Funcional.	144
Figura 59 – UCF - Integração Domínio eletrônica.	145
Figura 60 – Diagrama de integração lógica ET.	146
Figura 61 – Protótipo final EE.	147
Figura 62 – Protótipo final ET - Parte interna.	148
Figura 63 – Protótipo final ET - Parte externa.	149
Figura 64 – Pontos mapeados.	151
Figura 65 – Parametrização dos componentes físicos.	152
Figura 66 – Simulação das Zonas de Fresnel.	153
Figura 67 – Link de comunicação proposto.	154
Figura 68 – Link de comunicação - Vistas detalhadas.	154
Figura 69 – Padrão de radiação das antenas.	155
Figura 70 – Conversor USB-TTL / Radiotransmissor.	156
Figura 71 – Parametrização do Radiotransmissor.	156
Figura 72 – Instalação e operação EE.	157
Figura 73 – Posição Antena direcional ET.	158
Figura 74 – Display LCD - Inicialização do sistema.	158
Figura 75 – Display LCD - Transmissão do pacote.	158

Figura 76 – Display LCD - <i>TimeOut</i> e retransmissão do pacote.	159
Figura 77 – Display LCD - Confirma entrega do pacote.	159
Figura 78 – Instalações EE.	160
Figura 79 – Instalações ET.	160
Figura 80 – Análise da transação de pacotes.	161
Figura 81 – <i>Loop TimeOut</i> - ET.	162

Lista de tabelas

Tabela 1 – Definições do sistema de comunicação.	23
Tabela 2 – Distribuição das faixas de frequência de comunicação.	30
Tabela 3 – Faixas autorizadas para o Serviço de Radioamador por satélite.	32
Tabela 4 – Alcance de propagação por faixas.	34
Tabela 5 – Modelos de Referência.	38
Tabela 6 – Caixas de imersão da Matriz de Adoção.	49
Tabela 7 – Ferramentas de <i>software</i> para modelagem em equipe	55
Tabela 8 – Ferramentas para arquitetura e simulação de produtos	56
Tabela 9 – Necessidades dos usuários.	61
Tabela 10 – Requisitos do cliente.	62
Tabela 11 – Vínculo dos requisitos ao domínio cruzado..	65
Tabela 12 – Classificação e quantificação dos requisitos do sistema.	66
Tabela 13 – Lista de requisitos funcionais do sistema.	68
Tabela 14 – Especificação do requisito - [RQF01].	69
Tabela 15 – Especificação do requisito - [RQF02].	69
Tabela 16 – Especificação do requisito - [RQF03].	70
Tabela 17 – Especificação do requisito - [RQF04].	70
Tabela 18 – Especificação do requisito - [RQF05].	71
Tabela 19 – Especificação do requisito - [RQF06].	71
Tabela 20 – Especificação do requisito - [RQF07].	72
Tabela 21 – Especificação do requisito - [RQF08].	72
Tabela 22 – Especificação do requisito - [RQF09].	73
Tabela 23 – Especificação do requisito - [RQF10].	73
Tabela 24 – Especificação do requisito - [RQF11].	74
Tabela 25 – Especificação do requisito - [RQF12].	74
Tabela 26 – Especificação do requisito - [RQF13].	75
Tabela 27 – Especificação do requisito - [RQF14].	75
Tabela 28 – Especificação do requisito - [RQF15].	76
Tabela 29 – Especificação do requisito - [RQF16].	76
Tabela 30 – Especificação do requisito - [RQF17].	77
Tabela 31 – Especificação do requisito - [RQF18].	77
Tabela 32 – Especificação do requisito - [RQF19].	78
Tabela 33 – Especificação do requisito - [RQF20].	78
Tabela 34 – Especificação do requisito - [RQF21].	79
Tabela 35 – Especificação do requisito - [RQF22].	79
Tabela 36 – Especificação do requisito - [RQF23].	80

Tabela 37 – Especificação do requisito - [RQF24].	80
Tabela 38 – Lista de requisitos não-funcionais.	82
Tabela 39 – Especificação do requisito não-funcional - [RQNF01].	84
Tabela 40 – Especificação do requisito não-funcional - [RQNF02].	84
Tabela 41 – Especificação do requisito não-funcional - [RQNF03].	85
Tabela 42 – Especificação do requisito não-funcional - [RQNF04].	85
Tabela 43 – Especificação do requisito não-funcional - [RQNF05].	86
Tabela 44 – Especificação do requisito não-funcional - [RQNF06].	86
Tabela 45 – Especificação do requisito não-funcional - [RQNF07].	87
Tabela 46 – Especificação do requisito não-funcional - [RQNF08].	87
Tabela 47 – Especificação do requisito não-funcional - [RQNF09].	87
Tabela 48 – Especificação do requisito não-funcional - [RQNF10].	88
Tabela 49 – Especificação do requisito não-funcional - [RQNF11].	88
Tabela 50 – Especificação do requisito não-funcional - [RQNF12].	88
Tabela 51 – Especificação do requisito não-funcional - [RQNF13].	89
Tabela 52 – Modelos de Interfaces Microcontroladas.	94
Tabela 53 – Modelos de Radiotransmissores.	96
Tabela 54 – Modelos de Componentes Eletrônicos.	98
Tabela 55 – Modelos de Baterias.	99
Tabela 56 – Modelos de Antenas omnidirecionais.	103
Tabela 57 – Modelos de Antenas direcionais.	104
Tabela 58 – Modelo Caixa hermética.	105
Tabela 59 – Soluções Domínio Físico	112
Tabela 60 – Modelagem das variáveis.	118
Tabela 61 – Potência de recepção.	131

Lista de símbolos

Símbolos latinos

A	: Abertura física da antena	[m]
A_{ef}	: Área de abertura efetiva da antena de recepção	[m ²]
c	: Velocidade da luz	(m/s)
d	: Distância de separação T-R	[m]
d_1	: Distância para o transmissor	[m];
d_2	: Distância para o receptor, a partir de um ponto ao longo do caminho	[m]
f	: Frequência da portadora	[Hz]
G_r	: Ganho da antena receptora	[-]
G_t	: Ganho da antena transmissora	[-]
L	: Fator de perda do sistema não relacionado à propagação	[-]
PL	: Perda no espaço livre	[-]
P_r	: Potência recebida	[W]
P_t	: Potência transmitida	[W]
r	: Raio da primeira zona de Fresnel	[m];

Símbolos gregos

η	: Eficiência de abertura	[-]
λ	: Comprimento de onda	[m]
ϕ	: Fluxo de potência	[-]
π	: Proporção numérica	[-]
ω_c	: Frequência da portadora	[rad/s];

Siglas

ADC	<i>Architecture Context Diagram</i>
-----	-------------------------------------

ADCS	<i>Attitude Determination and Control Subsystem</i>
AOCS	<i>Attitude and Orbital Control System</i>
AM	Amplitude Modulada
ARM	<i>Advanced RISC Machine</i>
ASI	<i>Italian Space Agency</i>
CAD	<i>Computer Aided Design</i>
CAE	<i>Computer Aided Engineering</i>
COER	Certificado de Operador de Estação de Radioamador
COTS	<i>Comercial Off-the-shelf</i>
CPU	<i>Central Processing Unit</i>
CubeSat	<i>Cube Satellite</i>
DFA	<i>Architecture Flow Diagram</i>
DOU	Diário Oficial da União
EE	Estação Embarcada
EHF	<i>Extremely High Frequency</i>
ESA	<i>European Space Agency</i>
ET	Estação Terrestre
FBS	<i>Function-Behavior-Structure</i>
FCC	<i>Federal Communications Commission</i>
FDD	<i>Frequency-Division Duplexing</i>
FM	Frequência Modulada
FOOM	<i>Function & Object Mapping Model</i>
FOSAT	<i>Fiber Optic Sensing for Telecommunication Satellites</i>
FPGA	<i>Field Programmable Gate Array</i>
FT	<i>Function Trees</i>
GBs	<i>Gigabytes</i>

GHz	<i>Gigahertz</i>
GNSS	<i>Global Navigation Satellite Systems</i>
GPS	<i>Global Positioning System</i>
HF	<i>High Frequency</i>
HIL	<i>Hardware-In-the-Loop</i>
HOOM	<i>High Order Object Model</i>
HT	<i>Hand-Talk</i>
KBs	<i>Kilobytes</i>
IDE	<i>Integrated Development Environment</i>
IHM	Interface Homem-Máquina
IoT	<i>Internet of Things</i>
ISDN	<i>Integrated Services Digital Network</i>
ISM	<i>Industrial Scientific and Medical</i>
ITU	<i>International Telecommunication Union</i>
LAICA	Laboratório de Ciência e Inovação Aeroespacial
LAICAnSat	Projeto do Laboratório de Ciência e Inovação Aeroespacial
LAN	<i>Local Area Network</i>
LEO	<i>Low Earth Orbit Satellite</i>
LF	<i>Low Frequency</i>
LODESTAR	Laboratório de Simulação e Controle de Sistemas Aeroespaciais
MBD	<i>Model-Based Design</i>
MBs	<i>Megabytes</i>
MF	<i>Medium Frequency</i>
MHz	<i>Megahertz</i>
MIL	<i>Model-In-the-Loop</i>
MLI	<i>Multilayer Insulation</i>

NASA	<i>National Aeronautics and Space Administration</i>
OBDH	<i>On-board Data Handling</i>
ODCS	<i>Orbital Determination and Control Subsystem</i>
OO	<i>Object-oriented</i>
P2P	<i>Pear to Pear</i>
PIL	<i>Processor-In-the-Loop</i>
RF	Radiofrequencia
RENER	Rede Nacional de Emergência de Radioamadores
SHF	<i>Super High Frequency</i>
SIL	<i>Software-In-the-Loop</i>
SNR	<i>Signal-to-Noise Ratio</i>
T-R	Transmissor-Receptor
T&C	<i>Telemetry and Command</i>
TCS	<i>Thermal Control Subsystem</i>
TDD	<i>Time Division Duplex</i>
TDRSS	<i>Tracking and Data Relay Satellite System</i>
THF	<i>Tremendously High Frequency</i>
THz	<i>Terahertz</i>
TT&C	<i>Telemetry, Tracking, and Command</i>
UCF	<i>Unidade de Controle Funcional</i>
UHF	<i>Ultra High Frequency</i>
ITU	<i>International Telecommunication Union</i>
UnB	Universidade de Brasília
VHF	<i>Very High Frequency</i>

Sumário

1	INTRODUÇÃO	1
1.1	Objetivos	3
1.1.1	Objetivo Geral	3
1.1.2	Objetivos Específicos	3
1.2	Organização da dissertação	3
2	SATÉLITES DE PEQUENO PORTE	5
2.1	Evolução dos Satélites	5
2.1.1	<i>Lean Satellite</i>	6
2.2	CubeSats	7
2.2.1	Subsistemas	11
2.2.1.1	Subsistema de Fornecimento de Energia (EPS)	12
2.2.1.2	Subsistema de Propulsão	12
2.2.1.3	Subsistema de Controle de Atitude e Órbita (AOCS)	13
2.2.1.4	Subsistema de Processamento de Dados (OBDH)	13
2.2.1.5	Subsistema de Controle Térmico (TCS)	14
2.2.2	Telemetria, Rastreamento e Comando (TT&C)	15
2.2.2.1	Telemetria	15
2.2.2.2	Rastreamento	18
2.2.2.3	Comando	19
2.3	Comunicações via satélite	21
2.3.1	Elementos de um sistema de comunicação	21
2.3.2	Recursos primários e condições operacionais	23
2.3.3	Modelo de propagação no espaço livre	24
2.3.4	Espectro de frequências	28
2.4	Resumo do capítulo	35
3	PROCESSO DE DESENVOLVIMENTO DO SISTEMA	36
3.1	Metodologia	37
3.1.1	Abordagens para o desenvolvimento de sistemas	37
3.1.1.1	Compreensão do desenvolvimento do produto	38
3.1.1.2	Estrutura de decomposição orientada a objetos	39
3.1.1.3	Metodologia baseada em SysML	41
3.1.1.4	Modelo de Interface Multidisciplinar	41
3.1.1.5	Projeto Hierárquico	42
3.1.1.6	Modelo V	43

3.2	Modelo MBD (<i>Model-Based Design</i>)	44
3.2.1	Modelo tradicional versus MBD	45
3.2.2	Matriz de Adoção	47
3.2.3	Proposta de Integração das Metodologias: MBD e Modelo V	50
3.2.4	Técnicas adotadas para testes e verificações	53
3.2.5	Ferramentas de apoio a modelagem	55
3.2.5.1	<i>Software Matlab/Simulink®</i>	56
3.2.5.2	<i>Softwares Complementares</i>	57
3.3	Resumo do capítulo	59
4	MODELAGEM DO SUBSISTEMA DE COMUNICAÇÃO	60
4.1	Requisitos do subsistema de comunicação	60
4.1.1	Especificação dos requisitos funcionais	66
4.1.2	Especificação dos requisitos não-funcionais	82
4.2	Arquitetura do subsistema de comunicação	89
4.2.1	Arquitetura física	90
4.2.2	Mapa tecnológico - Domínio eletrônica	92
4.2.2.1	Interface Microcontrolada	94
4.2.2.2	Radiotransmissor	95
4.2.2.3	Componentes eletrônicos	97
4.2.2.4	Baterias	99
4.2.3	Mapa tecnológico - Domínio mecânica	100
4.2.3.1	Antenas omnidirecionais	102
4.2.3.2	Antenas direcionais	102
4.2.3.3	Unidade de Condicionamento Funcional	104
4.2.4	Arquitetura lógica	105
4.3	Arquitetura proposta - Integração dos domínios	112
4.4	Resumo do capítulo	114
5	TESTES, RESULTADOS E DISCUSSÕES	116
5.1	Fluxo de testes	116
5.1.1	Modelo Virtual	116
5.1.1.1	Bloco gerador de pulsos	119
5.1.1.2	Bloco radiotransmissor Tx	120
5.1.1.3	Bloco antena Tx e Rx	120
5.1.1.4	Bloco espaço livre	121
5.1.1.5	Bloco alvo	123
5.1.1.6	Bloco radiotransmissor Rx	124
5.1.1.7	Modelo proposto	125
5.1.2	Teste MIL (<i>Model-In-The-Loop</i>)	128

5.1.2.1	Análise do gerador de pulsos	129
5.1.2.2	Análise do espectro de frequência Tx e Rx	129
5.1.2.3	Análise do sinal Rx	131
5.1.3	Teste SIL (<i>Software-In-The-Loop</i>)	135
5.1.4	Teste HIL (<i>Hardware-In-The-Loop</i>)	136
5.1.4.1	Integração física - Estação Embarcada	137
5.1.4.2	Integração lógica - Estação Embarcada	139
5.1.4.3	Integração física - Estação Terrestre	141
5.1.4.4	Integração lógica - Estação Terrestre	146
5.1.4.5	Protótipo final	147
5.1.5	Testes práticos de comunicação	150
5.1.5.1	Análise do perímetro	151
5.1.5.2	Análise da diretividade	153
5.1.5.3	Análise prática	155
5.1.6	Considerações Finais	163
5.2	Resumo do capítulo	165
6	CONCLUSÕES	167
6.1	Proposta de Integração das Metodologias	167
6.2	Soluções dos Domínios Específicos	168
6.3	Modelo Virtual, Testes e Verificações	168
6.4	Concepção do Protótipo	169
6.5	Trabalhos Futuros	169
	REFERÊNCIAS	172
	APÊNDICES	179
	APÊNDICE A – DIAGRAMA DE MUDGE	180
	APÊNDICE B – TESTE SIL - CÓDIGO C	181
	APÊNDICE C – CÓDIGO FONTE EE	236
	APÊNDICE D – CÓDIGO FONTE ET	241
	APÊNDICE E – UNIDADE DE CONDICIONAMENTO FUNCIO- NAL (UCF-ET)	245

ANEXOS	247
ANEXO A – RESOLUÇÃO N° 697	248
ANEXO B – DATASHEET E32 433T30D	250
ANEXO C – DATASHEET GY450	255
ANEXO D – DATASHEET ATMEGA328	257

1 Introdução

Os avanços da tecnologia tornam cargas úteis e instrumentos para missões espaciais menores, mais leves e mais eficientes no consumo de energia, um nicho de mercado vem se consolidando dentro da comunidade acadêmica e já avança além das instituições para o setor comercial, viabilizando missões espaciais de baixo custo, em plataformas estratosféricas, classificadas como satélites de pequeno porte (TOORIAN ARMEN; DIAZ, 2008).

Neste contexto, surge a nomenclatura de *Lean Satellite*, na tradução para o português, Satélite Enxuto que utiliza abordagens de gerenciamento e desenvolvimento de riscos não tradicionais para obter entrega rápida e de baixo custo (CHO et al., 2015). Para alcançar esses dois pontos, o projeto do satélite conta com o uso de unidades comerciais não qualificadas para uso espacial e o tamanho do satélite torna-se inerentemente menor.

Inserido nesta classe, está o padrão CubeSat, termo acrônimo formado pela palavra cubo (do inglês *cube*) acrescida das três primeiras letras da palavra satélite. O termo é utilizado para designar um satélite de pequeno porte, em formato cúbico, cujas arestas medem 10 centímetros e que obedece a um padrão internacional em relação à estrutura, volume e massa, o qual é descrito por uma especificação de domínio público (PIGNATELLI; MEHRPARVAR, 2013).

Os CubeSats são, normalmente, desenvolvidos por meio de uma arquitetura aberta padronizada para os subsistemas envolvidos. O uso de módulos favorece o conceito de “containerização”, facilitando o desenvolvimento de cargas úteis (CGEE, 2016). Essa padronização simplifica a metodologia de testes, fornece flexibilidade de lançamento e envolve uma equipe multidisciplinar, composta por profissionais inseridos em seus domínios específicos e integrados a plataforma. As estruturas podem ser do tipo personalizadas ou comerciais (COTS do inglês, *Comercial Off-the-shelf*). A grande vantagem das estruturas COTS, está relacionada as suas simplicidades e herança de voo, enquanto estruturas personalizadas podem ser adotadas para atender a requisitos específicos de complexidade de missão, carga útil e subsistema, embora à custa de testes extensivos (POGHOSYAN; GOLKAR, 2017).

Neste cenário, iniciou-se, em 2013, no Laboratório de Simulação e Controle de Sistemas Aeroespaciais (LODESTAR) da Universidade de Brasília (UnB), hoje parte do Laboratório de Ciência e Inovação Aeroespacial (LAICA), o projeto LAICAnSat (BORGES et al., 2018), formado por um grupo multidisciplinar que tem entre suas atividades a construção de uma plataforma estratosférica, seguindo o padrão CubeSat, com o compromisso de permitir a evolução tecnológica em aplicações espaciais, oferecendo flexibilidade, confiabilidade, miniaturização e viabilidade financeira aos subsistemas agregados neste tipo

de interface, visando atender a desafios nos testes de pequenos satélites em altitudes compreendidas pela estratosfera, situada entre 7 Km e 50 Km de altitude. Aliado às questões tecnológicas, o projeto firma compromisso da divulgação científica das capacidades deste tipo de sistema para acomodar os requisitos de instrumentação, as soluções exclusivas abordadas por esta estrutura e o desenvolvimento de novas tendências e descobertas no âmbito das aplicações de baixa e alta altitude. Arelados a conceitos desafiadores de engenharia, que contribuem no entendimento dos subsistemas da plataforma e sua integração.

A abordagem dos CubeSats envolvendo o conceito de "containerização", projeto por módulos, apresenta um maior nível de risco de desempenho nos lançamentos, ou seja, os paradigmas de integração e desempenho ainda permanecem. Por meio da revisão da literatura foram identificadas sete abordagens adotadas no desenvolvimento de sistemas, sendo:

- a compreensão do desenvolvimento de produto, baseado no equilíbrio das áreas envolvidas (HEHENBERGER et al., 2011);
- a estrutura de decomposição orientada a objetos com foco na integração da equipe (WU et al., 2009);
- a metodologia baseada em SysML a partir de uma abordagem *top-down* (MHENNI et al., 2014);
- o modelo de interface multidisciplinar com foco no gerenciamento de compatibilidade das interfaces (ZHENG et al., 2016);
- o projeto hierárquico que separa as representações em conhecimento estrutural, comportamental ou funcional, buscando a integração das disciplinas (HEHENBERGER; ZEMAN, 2007);
- o modelo V que agrupa em seu processo elementos para envolvimento dos domínios específicos (GAUSEMEIER; MOEHRINGER, 2002);
- o modelo MBD (*Model-Based Design*), baseado na elaboração de modelos virtuais que proporcionam maior controle a equipe promovendo maior interação e confiança entre a simulação e prototipagem (PIETRUSEWICZ, 2019).

Para harmonia destes paradigmas este estudo preocupa-se em definir soluções relacionadas aos domínios específicos e validá-las através da elaboração de um modelo virtual, agregando condições que viabilizem a concepção para o desenvolvimento de um subsistema de comunicação, com ênfase para as funcionalidades de telemetria e comando. Neste contexto, a associação dos modelos MBD (PIETRUSEWICZ, 2019) e

V (GAUSEMEIER; MOEHRINGER, 2002), apresentam uma composição interessante, pois conseguem integrar por meio de uma metodologia híbrida, conceitos das etapas de desenvolvimento, testes, verificações e resultados, consistindo em uma condição de repetibilidade da metodologia, proporcionada pelo arranjo das etapas em composição as fases impostas pelo modelo híbrido, considerando aspectos do MBD e V, auxiliando em futuras replicações deste sistema, independente dos tipos de soluções envolvidas.

Este trabalho versa sobre o subsistema de Telemetria, Rastreamento e Comando (TT&C, do inglês *Telemetry, Tracking, and Command*), com ênfase nas funcionalidades de Telemetria e Comando, também conhecido como subsistema de comunicações, e tem como objetivo a adoção de uma proposta de metodologia para o desenvolvimento de uma interface de comunicação ponto-a-ponto do tipo *half-duplex* dedicado, aplicado a plataformas estratosféricas, em atendimento ao padrão CubeSat, voltada ao contexto do projeto LAICAnSat.

1.1 Objetivos

O presente trabalho buscará atingir os seguintes objetivos:

1.1.1 Objetivo Geral

O objetivo geral consiste em propor uma metodologia para desenvolver uma concepção de um subsistema de comunicação ponto-a-ponto dedicado a missões espaciais para testes estratosféricos, atendendo o padrão CubeSat.

1.1.2 Objetivos Específicos

- Definir a metodologia que será utilizada para o desenvolvimento do subsistema.
- Definir a arquitetura de um subsistema de comunicação para satélites de pequeno porte.
- Desenvolver e analisar os resultados referentes ao desenvolvimento do subsistema de comunicação.

1.2 Organização da dissertação

Este trabalho encontra-se estruturado de acordo com a seguinte distribuição de capítulos:

Capítulo 1 – INTRODUÇÃO: Apresenta o escopo do trabalho, os objetivos, a motivação e justificativa, direcionando a pesquisa dentro do universo acadêmico e científico.

Capítulo 2 – SATÉLITES DE PEQUENO PORTE: apresenta a fundamentação teórica sobre os conceitos que regem as plataformas estratosféricas, com destaque para o padrão CubeSat, sua arquitetura e subsistemas, com prioridade ao subsistema de Telemetria, Rastreo e Comando (TT&C), e ênfase às funcionalidades de Telemetria e Comando, aplicados como proposta de solução ao contexto atual do programa LAICAnSat, considerando à teoria da comunicação via satélite, com destaque para o modelo de propagação de ondas no espaço livre.

Capítulo 3 – PROCESSO DE DESENVOLVIMENTO DO SISTEMA: aborda a teoria de projeto de sistemas, orientando este tema em meio aos diversos modelos aplicados atualmente durante as etapas de desenvolvimento, projeto e concepção de sistemas. Cita quais métodos serão aplicados à metodologia deste trabalho e apresenta as técnicas de simulação em *software*, exploração das tecnologias mercadológicas adotadas e seu posicionamento no âmbito da pesquisa.

Capítulo 4 – MODELAGEM DO SUBSISTEMA DE COMUNICAÇÃO: apresenta a aplicação do processo proposto no desenvolvimento de sistemas para enlace de comunicação ponto-a-ponto do tipo *half-duplex* entre a estação embarcada e a estação terrestre, mapeando os requisitos e a arquitetura, posicionando em relação ao domínio específico.

Capítulo 5 – RESULTADOS E DISCUSSÕES: apresenta a análise de resultados, contribuições técnicas e considerações, relacionadas ao processo proposto através da construção de um modelo virtual, sua simulação e finalmente a concepção do protótipo físico.

Capítulo 6 – CONCLUSÃO: descreve a conclusão do trabalho, acerca dos objetivos atendidos, contribuições, limitações e sugestões para evolução nas pesquisas futuras e melhorias das funções de Telemetria e Comando integrantes do subsistema de comunicação.

2 Satélites de Pequeno Porte

Esta dissertação é de caráter multidisciplinar, em consequência, neste capítulo são apresentados os conceitos extraídos da revisão bibliográfica sobre sistemas espaciais e comunicação via satélite para o embasamento do estudo.

2.1 Evolução dos Satélites

O lançamento histórico do Sputnik 1 em 1957 marcou o início da era espacial (ROSCOSMOS, 2016). Posteriormente, nos últimos 60 anos centenas de satélites foram lançados para uma variedade de propósitos, como Ciências da Terra (BOARD et al., 2007), Astronomia e Astrofísica (BLANDFORD et al., 2010), Ciência Planetária (BOARD et al., 2012) e Heliofísica (NASA, 2015). Tradicionalmente, a indústria espacial produz grandes e sofisticadas espaçonaves fabricadas por equipes formadas por engenheiros e cientistas de diversas áreas do conhecimento. O desenvolvimento e lançamento de tais plataformas requer recursos significativos, ao alcance de apenas algumas grandes agências espaciais governamentais, como a Administração Nacional da Aeronáutica e Espaço (NASA, do inglês *National Aeronautics and Space Administration*) e a Agência Espacial Europeia (ESA, do inglês *European Space Agency*), entre outras.

A massa de lançamento dos satélites aumentou gradualmente e atingiu valores máximos no final dos anos 90 e início dos anos 2000, como evidenciados pelos satélites *Envisat Earth Observation* de 8,2 toneladas construído pela ESA e pela missão de exploração planetária Cassini de 5,7 toneladas construído pela NASA em colaboração com a ESA e a Agência Espacial Italiana (ASI), dados extraídos de JPL (2016). Essas grandes espaçonaves foram projetadas para reduzir o custo por kg de carga útil lançada e para aumentar as medições sinérgicas, incorporando múltiplos instrumentos em um único barramento de satélite. No entanto, interferências e impactos oriundos das micro vibrações, além de problemas de compatibilidade eletromagnética e diferentes níveis de maturidade dos instrumentos, proporcionaram problemas significativos de engenharia durante o desenvolvimento (DUBOCK et al., 2001). Assim, a atenção em satélites menores com conjuntos de instrumentos otimizados e leves (ou sensores únicos), aumentou na última década, tanto do ponto de vista operacional quanto de engenharia.

Avanços recentes na miniaturização da tecnologia permitiram que a indústria espacial construísse pequenos satélites a partir de componentes comercialmente prontos para uso, conhecidos pela sigla COTS, estes componentes apresentam baixo custo, baixa potência, além de compactos e confiáveis.

2.1.1 *Lean Satellite*

Os altos custos das missões espaciais baseadas em satélites, acentuados, principalmente, pelos custos de seus lançamentos, desencadearam uma corrida por melhorias na confiabilidade desses artefatos, com o intuito de garantir maiores condições de sucesso no objetivo das missões. Porém, o aumento da confiabilidade acarreta gastos financeiros e tempo de desenvolvimento ainda maiores, por exigir que tais sistemas ofereçam tecnologia de ponta e sejam submetidos a rigorosos testes buscando garantir o seu funcionamento no ambiente espacial da melhor forma possível, além de demandar conhecimento intelectual e envolvimento humano especializado.

Em função disso, o treinamento de recursos humanos para a área espacial, em especial de programas em centros de pesquisas e universidades, apresenta alguns problemas, já que esses altos custos e longos tempos de desenvolvimento praticamente impedem que uma missão espacial completa seja desenvolvida durante o período típico de uma formação em nível superior, nas áreas de Ciências Exatas e Engenharias. Para contornar esse problema, Jordi Puig-Suari, da *California Polytechnic State University*, e Bob Twiggs, da *Stanford University*, propuseram, em 1999, um modelo de satélite de pequeno porte que segue um padrão mais simples (TWIGGS, 2008). O intuito dessa ideia era fornecer, aos pesquisadores e alunos, a oportunidade de participar de um projeto espacial completo, incluindo a construção, os testes e a operação de um artefato com características similares aos primeiros satélites lançados.

Neste contexto, surge a nomenclatura de *Lean Satellite*, na tradução para o português, Satélite Enxuto, que utiliza abordagens de gerenciamento e desenvolvimento de riscos não tradicionais para obter entrega rápida e de baixo custo (CHO et al., 2015). Para alcançar esses dois pontos, o projeto do satélite conta com o uso de unidades comerciais não qualificadas para uso espacial e o tamanho do satélite torna-se inerentemente menor. O design aceita um certo nível de risco associado ao uso intensivo de componentes comerciais comuns (COTS). O número de membros da equipe também se torna menor. Além disso, as abordagens adotadas para estes satélites são diferentes das aplicadas para os satélites tradicionais, onde a confiabilidade frequentemente substitui o custo e o cronograma.

Dentro deste escopo, as aplicações de satélites enxutos apresentam uma solução que vai de encontro a questões como:

- Aceitação de peças e tecnologias COTS;
- Envolvimento de um grupo reduzido de profissionais;
- Curta duração das missões;
- Rápido desenvolvimento e obtenção de resultados em curto prazo;

- Minimização de resíduos;
- Mitigação do uso de materiais explosivos e/ou tóxicos;
- Simplificação do sistema funcional;
- Baixo controle de peças.

Estes tipos de satélites estão enquadrados na categoria de pequenos satélites e apresentam peso inferior a 180kg, sendo agrupados de acordo com a sua massa, com os minissatélites apresentando uma massa de 100-500kg, microssatélites com uma massa de 10-100kg, nanosatélites com uma massa de 1-10kg e picosatélites com massa abaixo de 1kg. A Figura 1 ilustra a classificação citada.

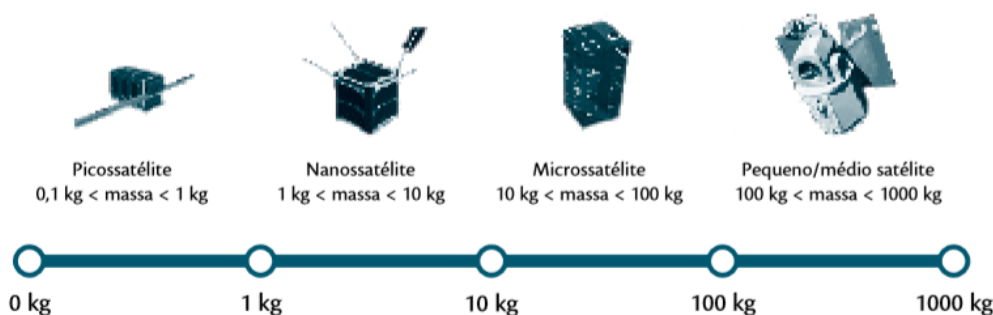


Figura 1 – Categorias dos pequenos satélites.
Fonte: CGEE (2016).

Esta tendência de miniaturização dos satélites inspirou o desenvolvimento de um novo conceito denominado CubeSat (PUIG-SUARI et al., 2001).

2.2 CubeSats

O padrão CubeSat foi criado pela *Stanford* e pela *California Polytechnic State Universities* em 1999 (PUIG-SUARI et al., 2001) e (BOUWMEESTER; GUO, 2010). O termo é um acrônimo formado pela palavra cubo (*cube*, em Inglês) acrescida das três primeiras letras da palavra satélite e é usado para designar um satélite de pequeno porte em forma de um cubo, cujas arestas medem 10 centímetros e que obedece a um padrão, o qual é descrito por uma especificação de domínio público (PIGNATELLI; MEHRPARVAR, 2013). Um cubo desses, ou uma unidade CubeSat (1U), tem um volume de um litro e sua carga útil pode ter massa de cerca de 1,3 quilogramas (kg). Essas unidades podem ser combinadas para formar satélites maiores (2U, 3U ou 6U, por exemplo).

Estes modelos foram inicialmente concebidos principalmente como plataformas de demonstração tecnológica ou educacional, segundo Poghosyan e Golkar (2017). No entanto,

foram recentemente desenvolvidas e propostas, missões mais avançadas envolvendo estas plataformas, indicando uma nítida transição da educação e tecnologia, demonstrando plataformas para missões reais de baixo custo com alto valor agregado em termos de retorno científico e receita comercial, conforme relata Selva e Krejci (2012).

O conceito de CubeSats desafia a espiral crescente de investimentos financeiros na área espacial ao adotar uma filosofia de aceitação de alguns riscos, com o uso intensivo de componentes comerciais comuns e otimização de testes, onde em função do atrativo que oferecem em termos de custo e tempo de desenvolvimento migraram rapidamente da academia para outros setores, incluindo o empresarial.

Estes cubos são o expoente de uma tendência de miniaturização dos satélites. Em muitos setores, o progresso tecnológico permitiu reduções significativas no volume ocupado pelos mais variados equipamentos, como em computadores e componentes eletrônicos em geral. Entretanto, só recentemente, com o advento da padronização, essa tendência começou a ser também observada em equipamentos espaciais (AGASID et al., 2015).

De certa forma, além de representarem uma inovação interessante na área espacial, eles correspondem a uma inovação no modelo de negócios a eles associados, uma vez que as maiores iniciativas estão fora do âmbito dos governos. Apenas recentemente, percebendo a importância dessas plataformas, algumas agências espaciais resolveram implantar ações relacionadas, sendo que em 2017, o número de lançamentos por ano deste tipo de plataforma superou o número de satélites convencionais, dado extraído de CGEE (2016).

Os CubeSats estão sendo desenvolvidos por meio de uma arquitetura aberta padronizada para os subsistemas mais comuns. O uso de módulos favorece o conceito de “containerização” e facilita o desenvolvimento de cargas úteis (CGEE, 2016). Essa padronização simplifica a metodologia de testes, fornece flexibilidade de lançamento e tem atraído a atenção dos mais variados setores, em diversos nichos de aplicações e de mercado. Além da padronização dos módulos, a padronização dos sistemas de ejeção em órbita é outro atrativo interessante, pois ajuda a diminuir custos e tempos de desenvolvimento de missões.

Estas plataformas possuem uma estrutura em formato cúbico, responsável por compor o chassi da plataforma e sua construção deve suportar mecanicamente todos os subsistemas do satélite, além de servir como blindagem térmica e de radiação para componentes sensíveis. As estruturas podem ser do tipo personalizadas ou comerciais (COTS). A grande vantagem das estruturas COTS, está relacionada as suas simplicidades e herança de voo, enquanto estruturas personalizadas podem ser adotadas para atender a requisitos específicos de complexidade de missão, carga útil e subsistema, embora à custa de testes extensivos (POGHOSYAN; GOLKAR, 2017).

Tipicamente, as estruturas CubeSat são feitas de alumínio, na pesquisa desenvolvida

por Agasid et al. (2015), mas estruturas impressas em 3D recentemente vêm despertando interesse entre os desenvolvedores, e várias missões já utilizam desta tecnologia, em destaque: 3U Tomsk-TPU-120 (TPU, 2016), 1U PrintSat (MSU, 2015), 2U QB50 UNSW EC0 (B.OSBORNE, 2015).

As principais características dos CubeSats são:

- compostos por unidades padronizadas cúbicas de 1U (10x10x10 cm), formando composições de 2U, 3U, 6U, etc;
- uso de sistemas de ejeção em órbita padronizados, denominados, por exemplo, P-POD (do inglês, *Poly Picosatellite Orbital Deployer*) ou SSPL (do inglês, *Space Shuttle Picosatellite Launcher*), capazes de liberar diversos satélites pela mesma interface. Existem sistemas comerciais destinados a satélites 1U, 2U, 3U e 6U; e
- uso de componentes COTS nos subsistemas de bordo.

A Figura 2, apresenta as configurações mais usuais para CubeSats.

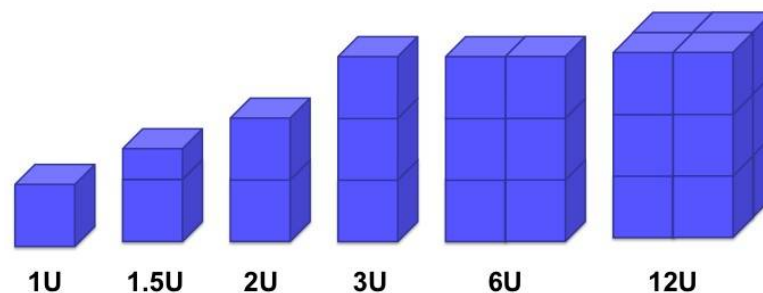


Figura 2 – Configurações para CubeSats.

Fonte: Initiative et al. (2017).

Normalmente, pequenos satélites são classificados com base apenas em sua massa, mas no caso do padrão CubeSat o volume também é considerado. A Figura 3 fornece uma classificação geralmente aceita para estas plataformas, juntamente com uma comparação com o padrão CubeSat (POGHOSYAN; GOLKAR, 2017), realizando uma relação entre as configurações já apresentadas nas Figuras 1 e 2.

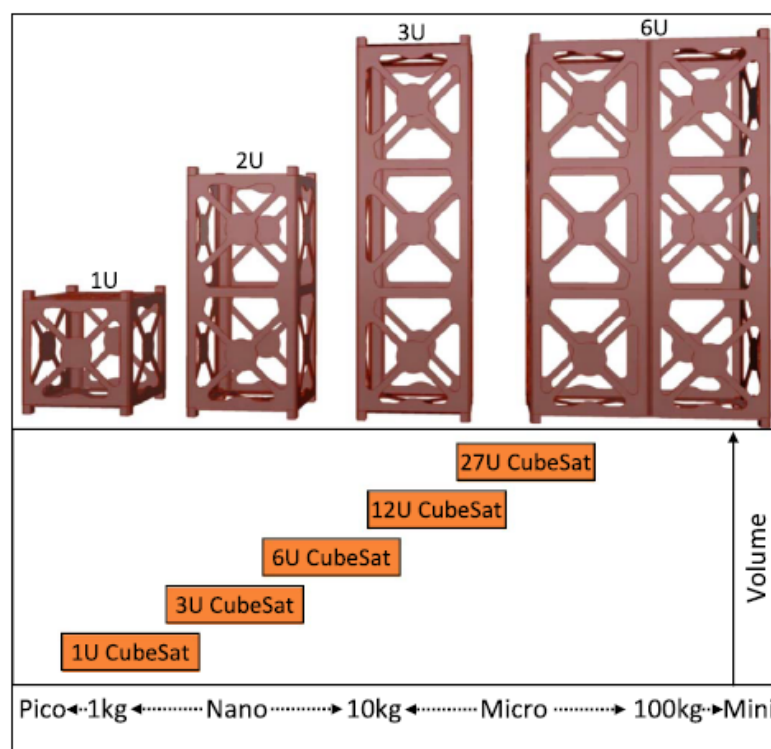


Figura 3 – Configurações satélite *versus* massa.
 Fonte: Poghosyan e Golkar (2017).

Originalmente, estes padrões de plataformas eram desenvolvidos principalmente por instituições acadêmicas e com apenas uma pequena fração de todos os lançamentos sendo pelo setor comercial, mas essa tendência vem mudando nos últimos anos e, embora as instituições de ensino ainda respondam por uma grande fração de todo o desenvolvimento e lançamento, a contribuição do setor comercial está aumentando significativamente (POGHOSYAN; GOLKAR, 2017). Devido ao fato desta solução despontar como uma inovação na área espacial, capaz de proporcionar às instituições e países que atualmente têm dificuldades de usar as aplicações espaciais em seu benefício, oportunidades concretas de acesso ao espaço para atender as suas demandas, conforme análise de (WOELLERT et al., 2011). Esses obstáculos se devem, principalmente, aos altos custos de uma missão espacial tradicional e à necessidade de contar com ampla infraestrutura de testes e equipes numerosas.

Um fator importante a ser considerado é que estas plataformas geram uma demanda constante junto às empresas. Embora os valores monetários sejam menores que os usuais do setor espacial, o fluxo contínuo de pedidos garante a manutenção de equipes e a sustentabilidade das instituições envolvidas, sendo um aspecto inovador capaz de promover uma mudança de paradigma no setor espacial, adequando-o à nova tendência de emprego de pequenos satélites para atender a diferentes tipos de exigências.

Dependendo da aplicação, um satélite deste porte pode ser completamente desen-

volvido e estar pronto para o lançamento em um período inferior a 18 meses, além de chegar a um custo abaixo de US\$ 100 mil. Essa enorme redução nos custos e no tempo de desenvolvimento possibilita ao setor espacial a exploração de novas estratégias e novos modelos de negócio.

De fato, já existem iniciativas direcionadas a fornecer respostas rápidas para o atendimento de demandas inesperadas, que necessitam de soluções espaciais, tais como as decorrentes de desastres naturais ou de situações de conflito. Por exemplo, em 2007, os Estados Unidos da América (EUA) criaram o *Operationally Responsive Space Office*, com o objetivo de assegurar o desenvolvimento de novas capacidades espaciais militares que pudessem rapidamente ser colocadas em operação. O atendimento desse tipo de demanda impõe uma nova lógica pertinente à aceitação de riscos e à confiabilidade de missões espaciais. Nesse sentido, os CubeSats vêm sendo considerados como uma solução altamente competitiva e que, em muitos casos, permite um equilíbrio aceitável entre as variáveis tempo, custo e confiabilidade.

2.2.1 Subsistemas

Os CubeSats seguem a composição dos satélites tradicionais, sendo compostos de vários subsistemas, os quais desempenham uma série de funções para cumprir a missão determinada (RODRIGUEZ, 2016). A complexidade dos subsistemas aumentou exponencialmente ao longo das últimas décadas. No entanto, a lógica funcional de cada subsistema, bem como aspectos da sua compatibilidade, não mudou significativamente (LEY et al., 2009). Na Figura 4 pode ser observada a distribuição da arquitetura que geralmente, compõe as plataformas dos satélites.

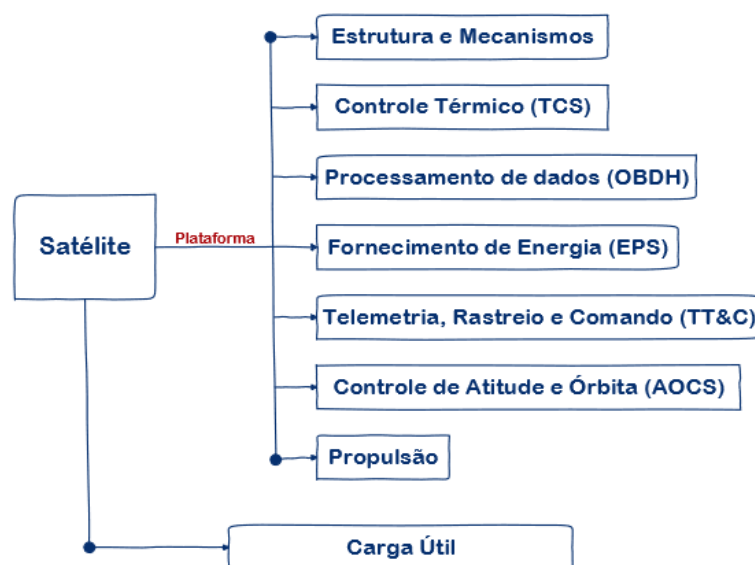


Figura 4 – Subsistemas dos satélites.
Fonte: Rodriguez (2016).

Cada subsistema apresenta uma função específica e envolve a composição de equipes multidisciplinares, para o seu desenvolvimento em atendimento às exigências do contexto da missão. A seguir será realizado uma breve explicação sobre cada um destes subsistemas, evoluindo até o subsistema foco desta dissertação, referente a Telemetria, Rastreo e Comando (TT&C), o qual comporá uma seção específica para um melhor detalhamento.

2.2.1.1 Subsistema de Fornecimento de Energia (EPS)

O subsistema de energia elétrica é assim constituído: uma fonte de energia, armazenamento de energia, distribuição de energia, unidades de controle e regulação (LARSON; WERTZ, 1992). As células solares fotovoltaicas são a principal fonte de energia para os CubeSats, com destaque para as células solares de junção tripla de última geração que podem obter eficiências entre 27 e 33% (AGASID et al., 2015).

Os sistemas de distribuição, regulação e controle de energia são frequentemente construídos por projetistas de naves espaciais com base em seus requisitos de sistema. Além disso, existem várias opções de sistemas prontos de gerenciamento de energia no mercado de CubeSat, fornecidos por empresas como a *Blue Canyon Technologies*, *Clyde Space* e *GomSpace* (POGHOSYAN; GOLKAR, 2017).

2.2.1.2 Subsistema de Propulsão

A capacidade de propulsão é crucial para aumentar as capacidades das missões dos futuros CubeSats, como a mudança e o levantamento de órbitas, a formação de voo, operações de proximidade, controle de atitude, ou arrasto e capacidade de reentrada no final da vida útil da missão, para atender aos requisitos de mitigação de detritos orbitais (JOHNSON; STANSBERY, 2010). Embora algumas dessas tarefas possam ser realizadas com dispositivos sem propulsão, suas aplicações são limitadas, aplicáveis principalmente a uma única tarefa e assumem seus próprios riscos (MUELLER et al., 2010).

Estes sistemas podem ser geralmente divididos em três categorias, incluindo sistemas químicos, elétricos e sem propulsão (MUELLER et al., 2010). Geralmente, os de propulsão químicos podem atingir maiores esforços de empuxo, embora com impulso específico limitado em comparação com os de propulsão elétrica (LARSON; WERTZ, 1992). Importante salientar que, sistemas sem propulsão, reduzem a complexidade e a massa do sistema, e potencialmente até mesmo podem viabilizar missões interplanetárias de longo prazo (AGASID et al., 2015).

Atualmente empresas como Busek Co. Inc., VACCO, Aerojet Rocketdyne, Sistemas Accion e Tethers Unlimited Inc. oferecem diferentes tipos de sistemas de propulsão para CubeSats.

2.2.1.3 Subsistema de Controle de Atitude e Órbita (AOCS)

O subsistema de Controle de Atitude e Órbita é formado pela combinação do Subsistema de Determinação e Controle de Orbitais (ODCS, do inglês *Orbital Determination and Control Subsystem*), responsável por medir e manter a posição do centro de massa do satélite em função do tempo e pelo Subsistema de Controle e de Determinação da Atitude (ADCS, do inglês *Attitude Determination and Control Subsystem*), responsável por medir e manter a orientação do satélite (LARSON; WERTZ, 1992). Na órbita da Terra, um Sistema Global de Navegação por Satélite (GNSS, do inglês *Global Navigation Satellite Systems*), GPS ou Galileo, é a principal técnica para a determinação de posição de satélites.

A ADCS usa sensores como rastreadores de estrelas, sensores solares, sensores da Terra e magnetômetros para determinar a atitude da espaçonave e usa atuadores como rodas de reação e propulsores para estabilizar e orientar satélites na direção desejada (MARTINEZ; PETRO, 2015).

O subsistema ADCS dos CubeSats melhorou drasticamente ao longo da última década, facilitado pelo desenvolvimento de rastreadores de estrelas miniaturizados de última geração, capazes de obter uma determinação de atitude de 3 eixos com uma precisão de segundos (MARTINEZ; PETRO, 2015). Além disso, várias empresas estão oferecendo unidades integradas para controle preciso de 3 eixos. Por exemplo, a *Blue Canyon Technologies* oferece um sistema integrado de controle de atitude, denominado CubeSat XACT, capaz de fornecer uma precisão de apontamento do satélite, com desempenho melhor do que $0,007^\circ$ para 3 eixos, ocupando apenas 0,5U do volume disponível (MASON et al., 2016).

2.2.1.4 Subsistema de Processamento de Dados (OBDH)

O subsistema de comando e manipulação de dados do satélite é responsável por receber, validar, decodificar e distribuir comandos para outros subsistemas, além de coletar, preparar e armazenar dados de manutenção e de missão para utilização de *downlink* ou de bordo (POGHOSYAN; GOLKAR, 2017). Em geral, o subsistema de comando e manipulação de dados também integra funções adicionais, como monitoramento da integridade do computador, interfaces de segurança e cronômetro de satélites (LARSON; WERTZ, 1992).

Avanços recentes em tecnologias de microcontroladores comerciais permitem capacidades de alto desempenho, embora com maior vulnerabilidade à radiação espacial. Os sistemas comuns de manuseio de dados *on-board* para o CubeSat incluem FPGAs (*Field Programmable Gate Array*) (BEKKER et al., 2011), microcontroladores MSP e PIC, bem como microcontroladores baseados em arquitetura ARM (*Advanced RISC Machine*) de

alto desempenho e eficiência energética (BOUWMEESTER; GUO, 2010) (MARTINEZ; PETRO, 2015). Além disso, plataformas promissoras de desenvolvimento de *software* e *hardware* de código aberto, como Arduino, BeagleBone e Raspberry Pi estão ganhando cada vez mais interesse entre os desenvolvedores CubeSat, essas tecnologias agregam alto valor, fornecendo ferramentas simplificadas e econômicas para pequenos desenvolvedores de satélites (MARTINEZ; PETRO, 2015).

O armazenamento de dados *on-board* do CubeSat pode ser tão baixo quanto vários KBs ou MBs e, dependendo dos requisitos da missão, a capacidade total de armazenamento pode ser aumentada até centenas de GBs, aproveitando as tecnologias comerciais de memória flash (ISIS, 2016) (TOORIAN ARMEN; DIAZ, 2008). O fator limitante fundamental para o CubeSat é o gargalo no *downlink* de dados e não o armazenamento de dados a bordo (SELVA; KREJCI, 2012).

2.2.1.5 Subsistema de Controle Térmico (TCS)

Em órbita, a espaçonave experimenta flutuações extremas de temperatura em curtos períodos de tempo (por exemplo, minutos a horas), exposta ao Sol a temperatura pode atingir mais de $+100^{\circ}\text{C}$, enquanto no eclipse a temperatura pode ficar bem abaixo de -100°C (LARSON; WERTZ, 1992). Assim, o controle térmico é crítico para o sucesso da operação e sobrevivência do satélite e sua carga útil.

Os controles de temperatura se apresentam como: controle térmico passivo e controle térmico ativo. Os controles do tipo passivo não utilizam entrada de energia e podem ser realizados por uma variedade de técnicas como isolamento de múltiplas camadas (MLI, do inglês *Multilayer Insulation*), revestimento térmico, protetores solares, tiras térmicas, grelhas, radiadores e tubos de aquecimento. Estes controles tem características que destacam como vantagens, relacionadas à confiabilidade, baixa massa, volume e custo, atendendo diretamente as exigências impostas pelo padrão CubeSat, dadas as restrições de potência, massa e volume (MARTINEZ; PETRO, 2015). Entretanto, sistemas do tipo ativo, que dependem de energia para operação, podem ser necessários para um controle térmico mais eficiente, durante missões que exijam faixas de temperaturas precisas, como refrigeração criogênica ou cargas biológicas (POGHOSYAN; GOLKAR, 2017).

Tradicionalmente, o isolamento térmico, como mantas MLI e revestimentos de superfície, é usado para regular o calor de entrada e evitar a dissipação excessiva de calor, a fim de manter os limites operacionais de temperatura dos subsistemas e em algumas cargas sensíveis, como cargas bioativas. O isolamento térmico tradicional pode ser combinado com sistemas de controle ativos para uma regulação térmica mais eficaz e precisa. Exemplos desta abordagem híbrida incluem Pharmasat (DIAZ-AGUADO et al., 2009) e BioSentinel (LEWIS et al., 2014) CubeSats. O protetor solar implantável é outro sistema de controle térmico passivo de última geração que está sendo desenvolvido no CryoCube-1 pela Sierra

Lobo, Inc. em colaboração com o Centro Espacial Kennedy da NASA (PUTMAN et al., 2015).

2.2.2 Telemetria, Rastreo e Comando (TT&C)

O subsistema TT&C é a interface entre o satélite e a estação terrestre, que permite o *downlink* dos dados de carga útil e manutenção para o centro de operações, e *uplink* através da transmissão de comandos do operador para a estação embarcada, bem como estabelecimento de comunicações entre satélites.

O objetivo da sua função é garantir a comunicação com o satélite durante a missão. Como parte da plataforma, este subsistema é necessário para todos os satélites, independentemente da aplicação (GUEST, 2016).

Como pode ser deduzido a partir do seu nome, segundo Guest (2016), este subsistema tem três funções específicas que devem ser executadas para garantir a capacidade do satélite, para atingir o sucesso da missão:

- **Telemetria:** A coleta de informações sobre a saúde e o status de todo o satélite e seus subsistemas e a transmissão desses dados para o segmento de comando no solo. Isso requer não apenas um sistema de telemetria na plataforma, mas também uma rede global de estações terrestres em todo o mundo para coletar os dados, a não ser, é claro, que a rede de satélites de aplicativos inclua links entre satélites capazes de transmitir os dados a um nó central;
- **Rastreamento:** O ato de localizar e seguir os satélites para permitir que o segmento de comando saiba onde o satélite está e para onde está indo. Novamente, isso requer um sistema de alcance na plataforma e uma rede de coleta de dados terrestre que permita que essa função de rastreamento funcione;
- **Controle:** A recepção e processamento de comandos para permitir a continuidade do funcionamento do satélite, a fim de fornecer o serviço de interesse. Novamente, um sistema terrestre se faz necessário.

Cada uma destas funcionalidades, são apresentadas a seguir, buscando um melhor detalhamento e compreensão do subsistema TT&C.

2.2.2.1 Telemetria

Telemetria é a coleção de medições e leituras de instrumento a bordo necessárias para verificar a saúde e o status de todos os subsistemas do satélite (BERTIGER et al., 1993). O subsistema TT&C deve coletar, processar e transmitir esses dados do satélite para o solo. O primeiro passo para fornecer atualizações de status ao solo é a coleta das

medições exigidas pelo segmento de comando. Medições relacionadas à saúde e status do satélite incluem:

- O status dos recursos (por exemplo, status de integridade e carregamento das baterias);
- A atitude do satélite (por exemplo, sistemas de rastreamento de radiofrequência);
- O modo de operação para cada subsistema (por exemplo, status de operação de um aquecedor, ligado ou desligado);
- A integridade de cada subsistema (por exemplo, saída dos painéis solares).

Essas medições não são necessárias apenas para o satélite, mas também para avaliar a saúde da carga útil. Em um satélite de comunicações, os dados de telemetria incluiriam informações como a configuração de comutação para o roteamento de sinais, a saída de energia dos transponders, a direção em que a antena é apontada ou o status dos sistemas de geração de imagens (GUEST, 2016). Todas essas medidas são coletadas com vários sensores, como termômetros, acelerômetros e transdutores, que fornecem saídas em formas como resistência, capacitância, corrente ou tensão medidas.

O projeto de uma plataforma orbital para a coleta de dados de tais sensores físicos, e a fiação associada necessária para reunir informações referentes ao status da estrutura e da carga útil, pode levar a um impacto de massa e custo. Por exemplo, algumas comunicações maiores ou satélites de sensoriamento remoto, podem ter até 500 sensores de temperatura a bordo da plataforma. A coleta de dados de um número tão elevado de sensores, pode levar a uma extensa cablagem (500 sensores de temperatura com 2 fios cada, resultaria em um total de 1000 fios). Isso levou a estudos relacionados à novas alternativas para a coleta de informações, como o projeto FOSAT (*Fiber Optic Sensing for Telecommunication Satellites*) da Agência Espacial Européia (ESA, do inglês *European Space Agency*), que usa fibra ótica em vez de fiação convencional para reunir dados mais eficientes sobre a saúde de um satélite (REUTLINGER et al., 2017).

O uso destes sensores para coletar as medições necessárias é apenas o primeiro passo no fornecimento de telemetria do satélite para a equipe de comando no solo. O segundo passo é o processamento das medições. Esse processamento inclui a conversão de medições analógicas em informações digitais, bem como a formatação de todas as medições para uma transmissão efetiva para a Terra. O processamento de dados de telemetria envolve dois fatores principais. Esses dois fatores envolvem a natureza dos padrões de automação da plataforma e os algoritmos de armazenamento de dados, os quais serão tipicamente diferentes para cada aplicação em particular e parâmetro da missão.

A automação refere-se especificamente à capacidade do satélite de interpretar e responder às medições de telemetria sem interação com o segmento de comando. Isso permite que o satélite emita comandos para os subsistemas diretamente. Normalmente pode ser encontrada em resposta a falhas previsíveis ou comuns em determinados subsistemas que exigem ações, como colocar componentes em espera quando operam fora de um intervalo de parâmetros. A automação permite ainda que ações instantâneas sejam tomadas a bordo de um satélite, este grau de operação autônoma pode variar amplamente entre os diversos satélites de aplicativos e a sofisticação do *software* embarcado.

A automação pode levar a três requisitos específicos para o subsistema TT&C, como observado abaixo (PISACANE, 2005):

- a capacidade do *software* integrado em identificar na telemetria se um subsistema está agindo incorretamente e, a capacidade correspondente de processar a resposta correta;
- a correta comunicação entre os componentes de comando e a passagem de informações dentro de um tempo pré-estabelecido;
- a capacidade de diagnóstico do *software*, ou seja, possibilitar que o sistema de telemetria determine se uma leitura inconsistente é ocasionada por outro subsistema ou por erros no próprio subsistema TT&C.

O armazenamento de dados de telemetria também pode ser necessário, pois as transmissões no solo podem não estar disponíveis a todo momento. Guest (2016) destaca ser extremamente caro estabelecer instalações terrestres suficientes para que um sistema global de satélites implantado em órbita terrestre baixa (LEO, do inglês *Low Earth Orbit Satellite*) tenha contato constante com a equipe de comando. A exceção seria no caso de existirem *links* entre satélites a bordo de todas as plataformas que permitiriam a retransmissão e comandos de telemetria. Devido à dificuldade de acesso contínuo, o *software* e os computadores de bordo devem ser capazes de processar e armazenar os dados dos sensores, enquanto aguarda a abertura de uma janela de comunicação. Os dados acumulados são informações essenciais para a análise da equipe de terra, particularmente para identificação de possíveis anomalias.

Uma alternativa ao armazenamento de dados é o uso de uma constelação de satélites que pode retransmitir a telemetria de qualquer satélite para locais específicos na Terra. Um exemplo é o Sistema de Satélite de Rastreamento e Retransmissão de Dados (TDRSS, do inglês *Tracking and Data Relay Satellite System*) da NASA, que consiste em nove satélites em órbita que transmitem as comunicações de qualquer outro satélite LEO para o seu segmento terrestre conhecido como o Complexo *White Sands* (IVANCIC, 2003).

O sistema de comunicações usado para telemetria de *downlink* pode ser o mesmo sistema usado para comunicar os dados de carga útil ou pode ser um sistema independente

dependendo da aplicação do satélite. As frequências típicas para o sistema de telemetria incluem: banda S (2,2 - 2,3 GHz), banda C (3,7 - 4,2 GHz) e banda Ku (11,7 - 12,2 GHz) (KEESE, 2003), outras bandas de frequência também podem ser empregadas para diferentes tipos de sistemas de satélite de aplicação. As comunicações de telemetria tendem a ter um erro de cerca de 10%, recomenda-se que sistemas de telemetria que utilizam frequências de banda Ku, façam alguma tolerância para a atenuação da chuva durante etapa de projeto do TT&C (LARSON; WERTZ, 1992).

2.2.2.2 Rastreo

Para se comunicar com um satélite, seja para receber telemetria ou transmitir comandos, o segmento de comando deve ser capaz de localizar e rastrear um satélite com precisão. Essas funções de alcance fazem parte da tarefa de rastreamento que é executada pelo subsistema TT&C. O satélite deve primeiro ser capaz de localizar e travar as transmissões entre a estação terrestre e o satélite. Quando o satélite é bloqueado, o subsistema TT&C determina o alcance, ou a distância da linha de visão entre a plataforma e sua velocidade radial, permitindo que o segmento de comando saiba a localização exata e o sentido do seu deslocamento (GUEST, 2016).

O processo de localização e bloqueio de um satélite a partir de uma estação terrestre é conhecido como rastreamento da transportadora. Este processo é mais comumente realizado em operações de satélite de aplicativo, usando um princípio conhecido como coerência de fase. O modo operacional típico deste princípio envolve o estabelecimento da relação entre a frequência de comunicação e a frequência de subida pré-determinada, permitindo a sincronização de suas fases.

Para determinar o alcance exato entre um satélite e uma estação terrestre, utiliza-se a aplicação de tons ou pseudocódigos. O tom ou código é modulado para a frequência de ligação ascendente e, quando o satélite o reconhece, o subsistema TT&C adiciona o mesmo tom ou código à ligação descendente. O segmento de comando pode então calcular o tempo de ida e volta necessário para esse tom, e usar essa informação para calcular a distância entre a estação terrestre e o satélite com a distância definida, a localização real do satélite pode ser determinada usando a informação apontada pela plataforma para determinar os ângulos de elevação e azimute (NASA, 2019).

O efeito Doppler é um meio alternativo para determinar alcance e velocidade radial do satélite, sendo melhor aplicado às plataformas em órbitas relativamente baixas. Consiste na mudança da frequência das transmissões causada pelo movimento relativo entre o transmissor e o receptor, conforme ilustrado na Figura 5, ou seja, quando o satélite está se aproximando de uma estação terrestre, a frequência que recebe é maior que a frequência transmitida, e vice-versa, valendo também para a frequência das transmissões que vão do satélite para a estação terrestre. Um problema com o uso do efeito Doppler

para determinação de localização, está relacionada a sempre existir dois locais, o local verdadeiro ou nominal e o local virtual ou espelhado, que são possíveis em um único ponto no tempo (ARGOS, 2016). Para dar conta disso, o subsistema TT&C precisa aplicar algoritmos de processamento para determinar qual o local correto. Satélites, como o Argos, usam dois algoritmos de posicionamento: análise de mínimos quadrados e filtragem de Kalman.

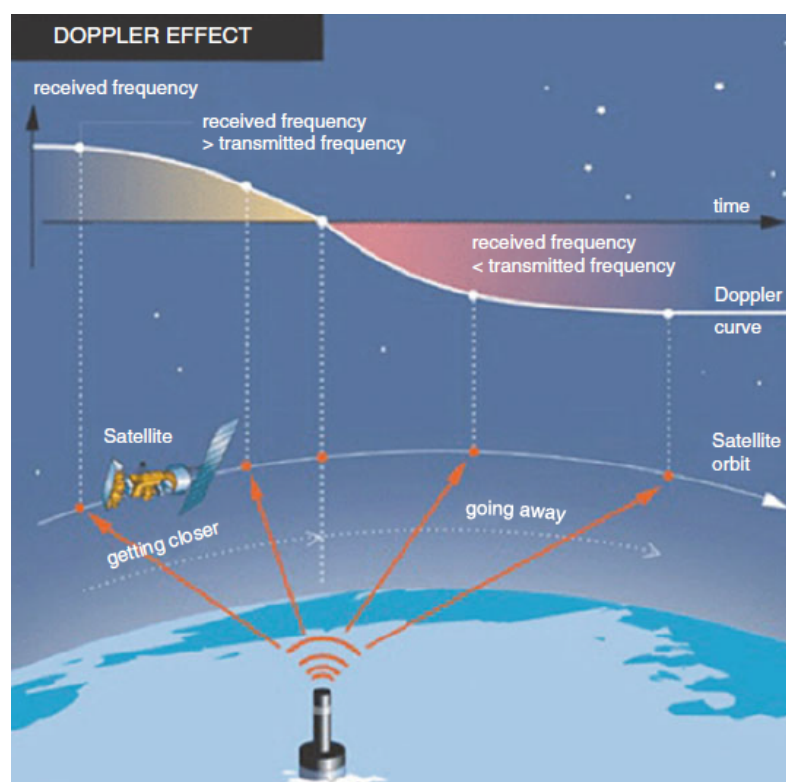


Figura 5 – Efeito Doppler - Rastreamento de satélites.
Fonte: ARGOS (2016).

2.2.2.3 Comando

Controle ou comando é a terceira função do subsistema TT&C, e é o ato de garantir os objetivos da missão espacial. Permitir o controle do satélite requer que o subsistema TT&C receba, processe e implemente os comandos exigidos pela estação terrestre. Alguns comandos podem ser automatizados através do uso de *software* integrado que implementa comandos pré-definidos mediante o reconhecimento de condições específicas. Alguns satélites projetados para “operação autônoma” levam esse grau de automação a níveis muito sofisticados.

Os comandos são usados para reconfigurar um satélite ou seus subsistemas para responder às condições da missão, podem incluir subsistemas de comutação e alterar condições de operações de componentes. De acordo com Guest (2016), também são utilizados para controlar a orientação e atitude da plataforma ou monitorar estruturas como matrizes solares ou antenas. Finalmente, os comandos podem vir na forma de programas

de *software* que são carregados no computador de bordo, através de microcontroladores para monitorar e atuar nos componentes continuamente, conforme necessidade.

O primeiro passo do sistema de comando é receber os dados do solo através do seu sistema de comunicação. As frequências de RF típicas dos sistemas de comando incluem: banda S (1,6-2,2 GHz), banda C (5,9-6,5 GHz) e banda Ku (14,0 14,5 GHz) (KEESE, 2003), outras bandas do espectro dentro das faixas VHF e UHF também são utilizadas, principalmente em comunicações do tipo ponto-a-ponto em visada direta. As taxas de dados típicas necessárias para sistemas de comando variam de 500 a 1.000 kb / s.

Uma vez que o satélite tenha recebido e demodulado as transmissões de comando de *uplink*, o sistema inclui três segmentos adicionais (GUEST, 2016):

- o decodificador de comando: responsável por reproduzir as mensagens de comando e produzir os sinais de bloqueio/habilitação e clock;
- a lógica de comando: responsável por validar a mensagem de comando e rejeitá-la em caso de inconformidade ou incoerência;
- e o circuito de interface: responsável por implementar a lógica de comando e se conectar aos demais sistemas.

O decodificador de comando coleta e processa todos os comandos recebidos de fontes originadas do solo e do computador de bordo, ele fornece atribuições de prioridade para cada comando dentro da fila de processamento. Devido à criticidade dos comandos de *uplink*, eles são frequentemente criptografados e, como observado acima, também podem exigir autenticação de outra instalação terrestre do TT&C.

Mensagens de comando típicas incluem *bits* validadores de entrada, *bits* de sincronização, *bits* de comando e *bits* de detecção de erro. O comando em si inclui o endereço do satélite e o tipo de comando. Em praticamente todos os sistemas de satélites de aplicações operacionais, possuem tarefas como: acionar um relé em um sistema, pulsar uma peça eletrônica, alterar o nível de saída de um componente, solicitar ou enviar dados para um componente.

A lógica de comando no subsistema TT&C deve verificar e validar o comando, garantindo que sejam enviados para o satélite corretamente. Uma vez que a lógica é usada para processar o comando, o subsistema TT&C ativa o circuito da interface conforme necessário. No caso de operações de resolução de problemas ou de recuperação de falhas, pode haver a necessidade de substituir restrições no *software* integrado para permitir que comandos que ofereçam maiores riscos sejam executados.

Esta dissertação relaciona-se diretamente com as funcionalidades de Telemetria e Comando do subsistema TT&C. Para uma melhor compreensão desses conceitos faz-se

necessário um estudo sobre as comunicações via satélite e os elementos dos sistemas de comunicação que estão envolvidos, além da forma de propagação usual neste tipo de comunicação. A próxima seção apresenta estas circunstâncias fornecendo uma direção em relação as comunicações via satélite e suas particularidades.

2.3 Comunicações via satélite

Em 1955, John R. Pierce propôs a utilização de satélites para comunicações. Esta proposta foi precedida, entretanto, por um artigo de Arthur C. Clarke, publicado em 1945, também propondo a idéia de utilizar um satélite orbitando a terra como ponto de repetição para a comunicação entre duas estações terrestres (HAYKIN; MOHER, 2009). O Sputnik I, foi lançado pela União Soviética em 1957, transmitindo sinais de telemetria durante 21 dias, um ano depois os Estados Unidos lançou o Explorer I, que conseguiu emitir sinais de telemetria no período de 5 meses, a partir de então as duas nações travaram uma corrida pela supremacia da tecnologia espacial, que chegou ao fim com a queda da União Soviética e o fim da Guerra Fria. Nesse período o satélite Telstar I lançado em julho de 1962, pelos Estados Unidos, trouxe um grande avanço na transmissão televisiva, sendo o primeiro satélite capaz de retransmitir programas de TV através do Atlântico.

Segundo Pelton et al. (2016), atualmente o mundo dos satélites pode ser dividido em duas grandes áreas, formadas pelos satélites científicos e os satélites de aplicação, que incluem: comunicações por satélite; radiodifusão por satélite; navegação, posicionamento e tempo de precisão por satélite; meteorologia geoestacionária e de órbita terrestre baixa; sensoriamento remoto e observação da Terra; e sistemas de informações baseados no espaço. Esta dissertação está restrita às comunicações por satélite, mais precisamente as funções de telemetria e comando integrantes do subsistema TT&C. As próximas seções fornecem sustentação em relação às formas básicas de comunicações, recursos primários e condições de operação que permeiam este tipo de comunicação.

2.3.1 Elementos de um sistema de comunicação

Duas formas básicas distinguem a maneira como é realizada a comunicação (HAYKIN; MOHER, 2009):

- *Broadcasting*: formada por um único e potente transmissor e vários receptores, com baixo custo. Nesta forma de sistema de comunicação, os sinais contendo a informação fluem para apenas uma direção, do transmissor para cada um dos receptores que se encontram na área de cobertura;
- Comunicação ponto-a-ponto (P2P, do inglês *Pear to Pear*): o processo de comunicação é realizado entre um único transmissor e um único receptor. Nesta outra forma existe,

geralmente, um fluxo bidirecional de sinais contendo informação, exigindo o uso de transmissor e um receptor (ou seja, um transceptor) em cada ponta do canal (Figura 6).

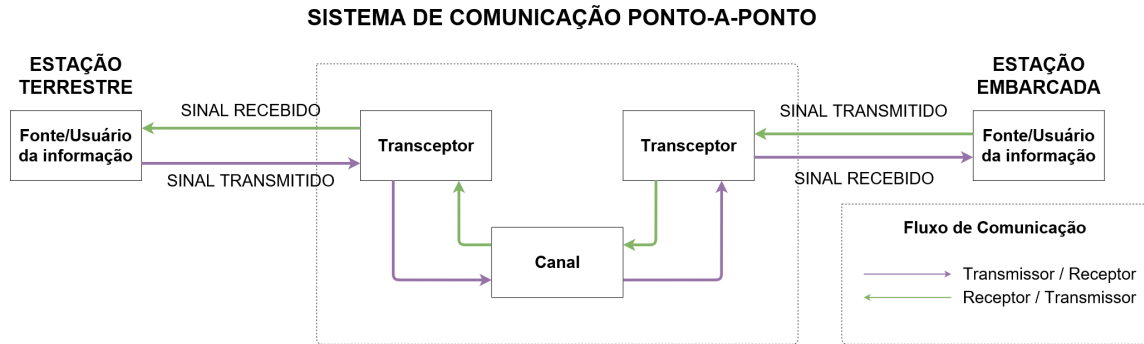


Figura 6 – Sistema de comunicação ponto-a-ponto. Adaptado de: Haykin e Moher (2009).

A forma de comunicação P2P simboliza o processo de um sistema de comunicação via satélite, formado pelos transceptores, canais de comunicação e entidades formadas pelas estações terrestres e embarcadas, que correspondem aos transmissores e receptores, conforme ilustra o diagrama de blocos da Figura 6.

Os sistemas de transmissão por RF podem ser classificados em *simplex*, semiduplex (em Inglês *half-duplex*) e duplex (em Inglês *full-duplex*) (Figura 7).

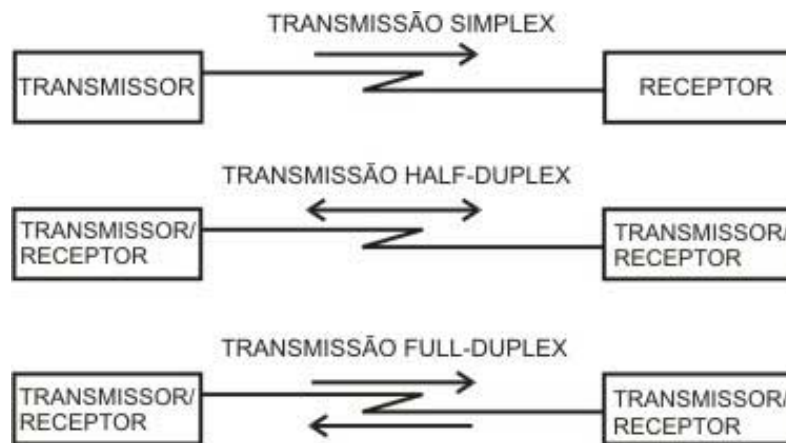


Figura 7 – Classificação dos sistemas de transmissão. Fonte: MFHRUG (2012).

Os sistemas simplex correspondem a mensagens recebidas, mas não confirmadas, exemplo: sistemas de *paging*. Os sistemas semiduplex ou *half-duplex*, permitem a comunicação bidirecional, mas usam o mesmo canal para transmissão e recepção, sendo permitida a transmissão ou recepção por vez, exemplo: sistemas de *walkie talkie*. Os sistemas duplex ou *full-duplex*, permitem a transmissão e recepção simultânea, pois oferece dois canais

simultâneos, mas separados, aplicados em sistemas duplex por divisão de frequência (FDD, do inglês *Frequency-Division Duplexing*), ou *slots* de tempo adjacentes em um único canal, aplicados em sistemas duplex por divisão de tempo (TDD, do inglês *Time Division Duplex*) (RAPPAPORT, 2009).

A partir dos conceitos de sistemas de comunicação descritos até o momento, um resumo das definições mais usuais dentro do escopo deste estudo para comunicações via satélite é apresentado na Tabela 1

Tabela 1 – Definições do sistema de comunicação.

SISTEMA DE COMUNICAÇÕES VIA SATÉLITE	
Estação Terrestre	Estação fixa, localizada em solo, que possui canais de rádio e antenas transmissoras e receptoras montadas em uma torre.
Canal de comando	Canal de rádio usado para transmissão de configuração, solicitação, e outras finalidades de orientação ou controle.
Canal direto	Canal usado para transmissão de informações da estação terrestre para a estação embarcada.
Sistemas simplex	Sistemas de comunicação que oferecem apenas comunicação unidirecional.
Sistemas semiduplex	Sistemas de comunicação que permitem a comunicação bidirecional usando o mesmo canal de rádio para transmissão e recepção. Em determinado momento, o usuário só pode transmitir ou receber informações.
Sistemas duplex	Sistemas de comunicação que permitem a comunicação bidirecional simultânea.
Estação Embarcada	Estação móvel, localizada em um satélite orbital ou não-orbital, que fecha o link de comunicação com a estação terrestre.
Canal reverso	Canal usado para transmissão da estação embarcada para a estação terrestre.
Transceptor	Um dispositivo capaz de transmitir e receber sinais de rádio simultaneamente.

Adaptado de: Rappaport (2009).

2.3.2 Recursos primários e condições operacionais

Os sistemas de comunicação são projetados para fornecer uma utilização eficiente de dois recursos primários de comunicação (HAYKIN; MOHER, 2009):

- Potência transmitida, definida como a potência média do sinal transmitido;

- Largura de faixa do canal, definida como o comprimento da faixa passante do canal.

Estes dois recursos definem os objetivos quanto a sua limitação e podem classificar os canais de comunicação em:

- Canal limitado em potência, onde o objetivo é a potência transmitida, como por exemplo: canais sem fio, canais de satélite e conexão de espaço profundo.
- Canais limitados em faixa, onde o objetivo é a largura de faixa do canal, como por exemplo: canais de telefonia e de televisão.

A limitação nos canais de satélite, é desejável por dois aspectos relevantes: o primeiro é viabilizar a potência transmitida o mais baixo possível para prolongar o tempo de uso da bateria do dispositivo, caso constatado em CubeSats que são alimentados por baterias externas, quando da não geração de energia por sistema fotovoltaico. O segundo é manter a potência transmitida em nível baixo do canal de descida, devido à limitação na potência disponível na estação embarcada, aspectos importantes nos requisitos do projeto desta dissertação, os quais são discutidos durante as etapas de modelagem do sistema no Capítulo 4.

Ainda de acordo com (HAYKIN; MOHER, 2009), outro ponto importante a se ter em mente nas condições operacionais dentro de um projeto de comunicação, é a presença inevitável de ruído na entrada do receptor, que tendem a perturbar a qualidade do sinal recebido. Uma forma quantitativa de contabilizar o efeito benéfico da potência transmitida em relação ao efeito degradante do ruído, é realizada em termos da relação sinal/ruído (SNR, do inglês *signal-to-noise ratio*). Definida como a razão da potência média do sinal recebido (ou seja, a saída do canal), pela potência média do sinal medido na entrada do receptor. Resultado expresso em decibel (dB).

O canal de RF impõe limitações que impactam o desempenho dos sistemas de comunicação sem fio. Segundo Rappaport (2009), o caminho do sinal entre o transmissor e o receptor varia desde a simples linha de visão até um que seja seriamente obstruído por prédios, montanhas e folhagens. A próxima sessão trata do modelo de propagação no espaço livre, modelo usual nos sistemas de comunicação por satélite e nos enlaces de rádio de microondas, devido suas características de enlace propiciar uma linha de visada direta entre transmissor e receptor.

2.3.3 Modelo de propagação no espaço livre

O modelo de propagação no espaço livre é usado para prever a intensidade do sinal recebido quando transmissor e receptor possuem um caminho de linha de visão limpo, desobstruído entre eles. O sistema de comunicação por satélite possui esta característica,

que tem como consequência, um decréscimo da potência recebida como uma função da distância de separação T-R elevada a alguma potência (RAPPAPORT, 2009).

O entendimento deste conceito é embasado por equações que fornecem dados para qualificação e dimensionamento do espaço entre um transmissor e receptor. Inserido neste contexto, está a potência no espaço livre recebida (P_r) por uma antena receptora que esta separada de uma antena transmissora, irradiando, por uma distância d , sendo representada pela equação do espaço livre de Friis

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L} \quad (2.1)$$

em que:

- $P_r(d)$: Potência recebida, função da separação T-R (W);
- P_t : Potência transmitida (W);
- G_t : Ganho da antena transmissora (-);
- G_r : Ganho da antena receptora (-);
- λ : Comprimento de onda (m);
- d : Distância de separação T-R (m);
- L : Fator de perda do sistema não relacionado à propagação ($L \geq 1$) (-).

as perdas variadas $L(L \geq 1)$, costumam ser provenientes da atenuação da linha de recepção, perdas de filtro e perdas da antena no sistema de comunicação. Valor para $L = 1$, sinaliza a ausência de perdas no *hardware* do sistema,

Haykin e Moher (2009) enfatiza, a necessidade do uso de uma antena altamente direcional para que a potência transmitida seja irradiada, particularmente, ao longo de uma direção específica de interesse. Portanto, para uma potência total (P_t), alimentando uma antena sem perdas com ganho (G_t), a densidade de fluxo de potência em uma distância d na direção do lóbulo principal da antena é dada por

$$\phi = \frac{P_t G_t}{4\pi d^2} \quad (2.2)$$

em que:

- ϕ : Fluxo de potência (-);
- P_t : Potência transmitida (W);
- G_t : Ganho da antena transmissora (-);

- d : Distância de separação T-R (m).

Segundo BALANIS (2009), a cada antena podemos associar um número de áreas equivalentes. Estas áreas são usadas para descrever as características de recepção da antena quando uma onda incide sobre ela. Uma destas áreas equivalentes diz respeito à área de abertura efetiva da antena de recepção, sendo representada pela fórmula

$$A_{ef} = \eta A \quad (2.3)$$

em que:

- A_{ef} : Área de abertura efetiva da antena de recepção (m²);
- η : Eficiência de abertura (-);
- A : Abertura física da antena (m).

tipicamente, η está na faixa percentual de 40 a 90, de acordo com o tipo de antena adotado. Ainda de acordo com BALANIS (2009), o ganho de uma antena, embora esteja relacionado a diretividade, é uma medida que leva em consideração tanto a eficiência como as propriedades direcionais da antena, para o caso de uma antena de recepção G_r , este pode ser definido em termos da abertura efetiva a partir das características de incidência da onda, de acordo com

$$G_r = \frac{4\pi A_{ef}}{\lambda^2} \quad (2.4)$$

em que:

- G_r : Ganho da antena de recepção (-);
- A_{ef} : Área de abertura efetiva da antena de recepção (m²);
- λ : Comprimento de onda (m).

A abertura efetiva A_{ef} está relacionada ao tamanho físico da antena e λ está relacionado à frequência da portadora por meio de

$$\lambda = \frac{c}{f} = \frac{2\pi c}{\omega_c} \quad (2.5)$$

em que:

- f : Frequência da portadora (Hz);
- ω_c : Frequência da portadora (rad/s);

- c : Velocidade da luz (m/s).

Um tipo particular de atenuação em transmissões via satélite e que interfere na qualidade do sinal é a perda no espaço livre ou perda do caminho (*path loss*), sua principal característica é o espalhamento do sinal e sua consequente atenuação à medida que a distância entre transmissor e receptor aumenta. Rappaport (2009) aponta que a perda no espaço livre representa a atenuação do sinal como uma quantidade positiva, sendo definida como a diferença entre a potência transmitida efetiva (P_t) e a potência recebida (P_r), e pode ou não incluir o efeito dos ganhos da antena. A perda do caminho para o modelo espacial quando os ganhos da antena são incluídos é dada por

$$PL(dB) = 10 \log \frac{P_t}{P_r} = -10 \log \left[\frac{G_t G_r \lambda^2}{(4\pi)^2 d^2} \right] \quad (2.6)$$

em que:

- $PL(dB)$: Perda no espaço livre (PL);
- $P_r(d)$: Potência recebida, função da separação T-R (W);
- P_t : Potência transmitida (W);
- G_t : Ganho da antena transmissora (-);
- G_r : Ganho da antena receptora (-);
- λ : Comprimento de onda (m);
- d : Distância de separação T-R (m).

Quando os ganhos da antena são excluídos, as antenas são consideradas como tendo ganho unitário, e a perda do caminho é dada por

$$PL(dB) = 32,5 + 20 \log d + 20 \log f \quad (2.7)$$

A propagação em espaço livre depende do caminho de visada direta entre o transmissor e o receptor, além de uma área desprovida de obstáculos ao longo do caminho. As comunicações via satélite experimentam este tipo de visada ideal, porém situações como chuvas, nuvens, além de fatores como interferências de outras ondas de RF concorrentes podem atrapalhar e proporcionar períodos de instabilidade (falta de comunicação) ou adição de ruídos ao canal de comunicação. A Figura 8 apresenta uma comunicação terrestre e uma simulação do espectro de diretividade entre as antenas, denominado *zona de Fresnel*, a qual define um elipsóide de revolução, em que objetos situados dentro da primeira zona de Fresnel afetarão a transmissão e causarão desvios do modelo de propagação em espaço

livre (HAYKIN; MOHER, 2009). O raio da primeira zona de Fresnel depende da posição entre a antena de transmissão e recepção, sendo dado por

$$r = \sqrt{\frac{\lambda d_1 d_2}{d_1 + d_2}} \quad (2.8)$$

em que:

- r : Raio da primeira zona de Fresnel (m);
- λ : Comprimento de onda da transmissão (m);
- d_1 : Distância para o transmissor (m);
- d_2 : Distância para o receptor, a partir de um ponto ao longo do caminho (m).

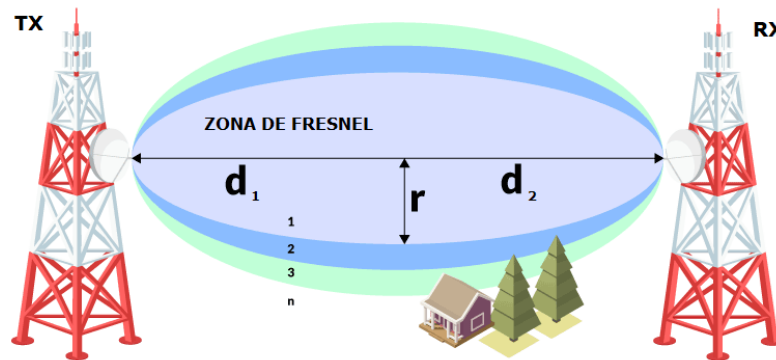


Figura 8 – Zona de Fresnel.
Adaptado de: Martínez (2018).

Importante abordar a existência da *zona de Fresnel*, para uma melhor compreensão do diagrama de radiação da antena e sua diretividade em termos de visada direta, entre o transmissor e o receptor.

2.3.4 Espectro de frequências

O sistema de comunicação via satélite, ou qualquer outro sistema que envolve transmissor e receptor, utiliza como meio de propagação, ondas eletromagnéticas. O número de oscilações por segundo de uma onda eletromagnética é chamado de frequência, f , e é medida em Hz. A distância entre dois pontos máximos (ou mínimos) consecutivos é chamada de comprimento de onda, e é representado pela letra grega λ (lambda) (TANENBAUM, 2003).

O princípio das comunicações sem fio, com destaque para a comunicação via satélite, é baseada no aspecto da transmissão de ondas eletromagnéticas dentro de um sistema de antenas transmissoras e receptoras separadas por uma determinada distância,

que atenda aos padrões para o qual foi projetada, buscando a transmissão eficiente da informação desejada. Ondas eletromagnéticas carregam consigo um determinado volume de informações que é limitado pela sua largura de banda. BALANIS (2009) posiciona que a largura de banda está diretamente relacionada à antena e pode ser definida como a faixa de frequências na qual o desempenho da antena, referido a algumas características, atende um padrão específico. Estes padrões podem estar vinculados às características como: impedância de entrada, diagrama, largura do feixe, polarização, nível de lóbulo secundário, ganho, direção do feixe e eficiência de radiação, sendo expressa de forma diferente para cada modelo de antena aplicado.

A velocidade, durante o caminho realizado por uma onda eletromagnética, considerando o vácuo, se mantém constante independente da frequência, e viaja na velocidade da luz, c , aproximadamente 3×10^8 m/s, mudanças no meio de propagação diminuem a intensidade desta velocidade. Estas grandezas se relacionam de acordo com a Equação (2.5).

O espectro eletromagnético é apresentado na Figura 9, permitindo uma análise das bandas e suas aplicações nos serviços de comunicações. A nomenclatura das frequências listadas na parte inferior da Figura 9, foram definidas pela ITU (*International Telecommunication Union*), agência responsável por padronizar e regular as ondas de rádio e telecomunicações internacionais.

Segundo Agasid et al. (2015) tradicionalmente, a comunicação entre a Terra e o satélite é baseada no espectro de rádio (de 3MHz a 40GHz). As diferentes faixas de comunicação definidas por Bruder et al. (2003) que são tipicamente usadas para satélites foram extraídas de forma macro a partir da Figura 9 e são apresentadas com maior exatidão na Tabela 2, partindo do espectro HF até o SHF, na banda Ka, além da faixa de comunicação óptica.

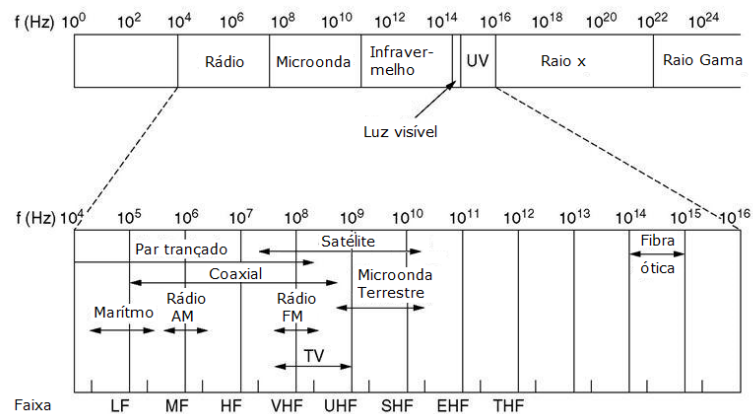


Figura 9 – O espectro eletromagnético e suas aplicações na comunicação. Adaptado de: Tanenbaum (2003).

A ITU divide o globo terrestre em três regiões, conforme o mapa da Figura 10, para fins de administração do espectro de radiofrequências. As administrações são orienta-

Tabela 2 – Distribuição das faixas de frequência de comunicação.

FAIXAS DE FREQUÊNCIAS DE COMUNICAÇÃO (TERRA-SATÉLITE)	
<i>High Frequency (HF)</i>	3 - 30 MHz
<i>Very High Frequency (VHF)</i>	30 - 300 MHz
<i>Ultra High Frequency (UHF)</i>	300 MHz - 3 GHz
Banda L	1 - 2 GHz
Banda S	2 - 4 GHz
Banda C	4 - 8 GHz
Banda X	8 - 12 GHz
Banda Ku	12 - 18 GHz
Banda K	18 - 27 GHz
Banda Ka	27 - 40 GHz
<i>Optical (Laser Communication)</i>	100 - 800 THz

Adaptado de: Agasid et al. (2015).

das a acompanhar as atribuições definidas para as faixas de frequências, aprovadas em Assembleias, por representantes dos países membros (ANATEL, 2018b). Na Figura 10, a região 2 é constituída pelas administrações dos países das Américas, entre os quais está o Brasil.

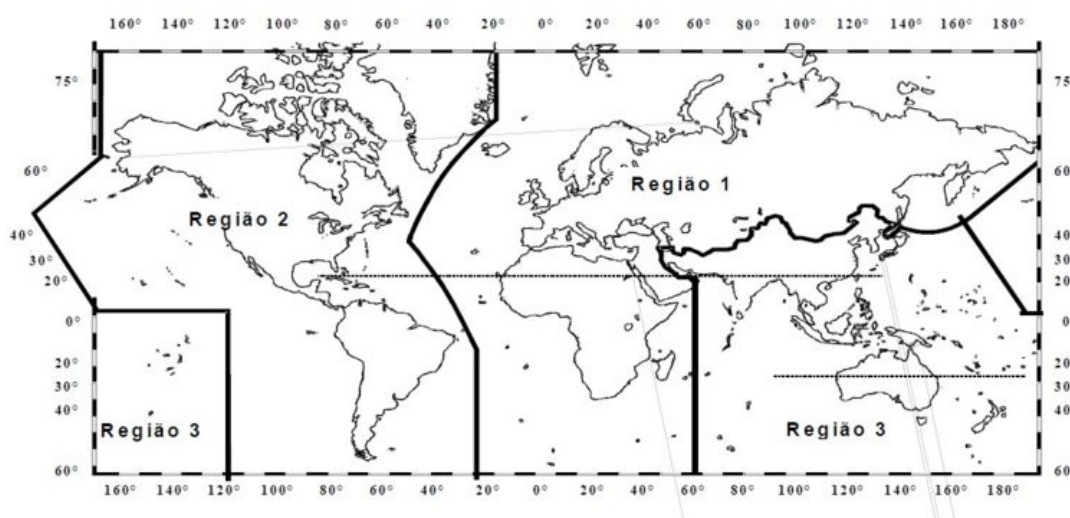


Figura 10 – Regiões de Administração do Espectro de RF.
Adaptado de: ANATEL (2018b).

A atribuição de faixas de frequência no Brasil, é encontrada no site da ANATEL (2018b), disponível nas referências deste estudo, local em que estão disponibilizadas todas as informações relacionadas aos serviços terrestres ou espaciais de radiocomunicação ou ao

serviço de radioastronomia, nos moldes do Regulamento de Radiocomunicações (RR) da ITU.

Kara et al. (2015) aponta, que para pequenos satélites a seleção de um espectro de frequência depende de vários fatores, como taxa de transferência de dados esperada, energia disponível, massa e problemas de licenciamento. Pensando em termos da tecnologia de transmissão apropriada para pequenos satélites, os transmissores *VHF/UHF* tem uma taxa de transferência máxima de aproximadamente 38 Kbps, os transmissores de banda *S* têm uma taxa de transferência máxima de 10 Mbps, transmissores de banda *X* em torno de 500 Mbps e bandas *K/Ku/Ka* em torno de 1,2 Gbps (AGASID et al., 2015).

As faixas de radiofrequência que compõem o espectro autorizado para uso do Serviço de Radioamador e por determinação também aplicadas ao radioamador por satélite, estão presentes na Resolução n° 697, de 28 de agosto de 2018, publicado do Diário Oficial da União (DOU) em 30 de agosto do mesmo ano, presente no Capítulo II, Artigo 3° (Anexo A). Inserido nesta resolução estão as faixas citadas na Tabela 4, a qual embasou as distâncias de propagação do sinal em condições ideais para análise dos intervalos de frequência, que são adotados nesta pesquisa.

Um ponto importante regido no Capítulo 2, Artigo 4° (Anexo A), desta resolução que regulamenta os limites gerais de potência por classe do COER (Certificado de Operador de Estação de Radioamador), nos incisos I, II e III, além dos casos de limites diferenciados para determinadas faixas de radiofrequência devido condições específicas, incluindo a convivência com outros serviços de radiocomunicações, em destaque no Artigo 5° deste mesmo Capítulo, incisos I, II, III e IV (Anexo A).

A Tabela 3, apresenta às faixas de frequência alocadas ao Serviço de Radioamador por satélite no Brasil e sua distribuição, considerando as faixas para HF, VHF, UHF, Banda S, Banda X e Banda K, destinadas para este tipo de serviço, segundo a Resolução n° 697, de 28 de agosto de 2018, Artigo 5°, parágrafo único e Artigo 6°, incisos 1° e 2° (ANATEL, 2018c):

Tabela 3 – Faixas autorizadas para o Serviço de Radioamador por satélite.

COMPRIMENTO DE ONDA	HF FAIXA	CARÁTER DE UTILIZAÇÃO
$\lambda = 40$ m	7,000-7,100 MHz	Primário
$\lambda = 20$ m	14,000-14,250 MHz	Primário
$\lambda = 17$ m	18,068-18,168 MHz	Primário
$\lambda = 15$ m	21,000-21,450 MHz	Primário
$\lambda = 12$ m	24,890-24,990 MHz	Primário
$\lambda = 10$ m	28,000-29,700 MHz	Primário
VHF		
$\lambda = 2$ m	144,000-146,000 MHz	Primário
UHF		
$\lambda = 70$ cm	430,000-440,000 MHz	Secundário
$\lambda = 23$ cm	1,260 - 1,270 GHz	Secundário
UHF E BANDA S		
$\lambda = 13$ cm	2,400 - 2,450 GHz	Secundário
BANDA S		
$\lambda = 9$ cm	3,400 - 3,410 GHz	Secundário
BANDA C		
$\lambda = 5$ cm	5,650 - 5,670 GHz	Secundário
	5,830 - 5,850 GHz	Secundário
BANDA X		
$\lambda = 3$ cm	10,450 - 10,500 GHz	Secundário
BANDA Ka		
$\lambda = 1,2$ cm	24,000 - 24,050 GHz	Primário

Fonte Própria.

Em análise a Tabela 3, a coluna Caráter de Utilização, especifica o nível de utilização de radiofrequências, segundo a Resolução nº 671, de 3 de novembro de 2016 (ANATEL, 2016), sendo conceituada da seguinte forma:

- uso em caráter primário: uso de radiofrequências caracterizado pelo direito à proteção contra interferência prejudicial;
- uso em caráter secundário: uso de radiofrequências caracterizado pelo direito à proteção contra interferência prejudicial, exceto quando proveniente do uso em caráter primário, ou uso subsidiário de radiofrequências associado a contrato de exploração industrial;
- uso exclusivo: hipótese em que uma autorização confere ao interessado o direito de utilizar-se de uma radiofrequência, faixa ou canal de radiofrequências, sem compartilhamento e em caráter primário, numa determinada área geográfica, durante um determinado período de tempo; e,
- uso não exclusivo: hipótese em que uma autorização confere ao interessado o direito de utilizar-se de uma radiofrequência, faixa ou canal de radiofrequências, com

compartilhamento e em caráter primário ou secundário, na mesma área geográfica.

O Ato nº 9106, de 22 de novembro de 2018, publicado no Boletim de Serviço Eletrônico em 26 de novembro do mesmo ano, regulamenta o benefício de operação para o Serviço de Radioamador brasileiro, buscando viabilizar a rádio experimentação e a operação em faixas de radiofrequência padronizadas internacionalmente, com destaque para as apresentadas na Tabela 3, este ato tem como finalidade resolver as seguintes situações, conforme exposto em ANATEL (2018a):

1. Publicar a lista de características básicas de emissão;
2. Publicar o plano de faixas com aplicações do Serviço de Radioamador;
3. Publicar a canalização de radiofrequências para estações repetidoras de fonia;
4. Publicar a canalização de radiofrequências para estações IVG (*Internet Voice Gateway*).

No que rege as faixas apresentadas na Tabela 3, suas frequências de operação (inicial e final), tipos de serviços, modos de operação, aplicações e observações, o ato nº 9106, regulamenta em seu Anexo B o plano de faixas com aplicações do Serviço de Radioamador por satélite, com mais precisão para as seguintes seções: espectro de HF, seções B.3.6 (Faixa dos 40 m), B.3.8 (Faixa dos 20 m), B.3.9 (Faixa dos 17 m), B.3.10 (Faixa dos 15 m), B.3.11 (Faixa dos 12 m) e B.3.12 (Faixa dos 10 m). Para o espectro de VHF a Seção B.3.14 (Faixa dos 2 m). Na faixa de UHF as seções B.3.16 (Faixa dos 70 cm) e B.3.18 (Faixa dos 23 cm). Para o espectro de congruência do UHF e da banda S destaque para a Seção B.3.19 (Faixa dos 13 cm). A faixa da banda S é regida pela Seção B.3.20 (Faixa dos 9 cm). Na banda C a Seção B.3.21 (Faixa dos 5 cm). A banda X regida pela Seção B.3.22 (Faixa dos 3 cm) e por fim a Seção B.3.23 (Faixa dos 1,2 cm) para a banda Ka. Todas estas faixas estão descritas em ANATEL (2018a) para esclarecimentos e entendimento mais aprofundado das aplicações, observações e notas específicas de operação.

Satélites universitários denominados CubeSats, por recomendação da ITU, frequentemente operam em frequências de rádio alocadas ao Serviço de Radioamador e assim devem cumprir as premissas legais que regulamentam este tipo de serviço. A Portaria nº 307, de 22 de julho de 2009, estabelece condições de ativação e execução da Rede Nacional de Emergência de Radioamadores - RENER, vale destacar pontos importantes desta portaria que recai sobre as faixas de operação que podem ser aplicadas aos CubeSats.

O capítulo 7, Seção 7.1, desta portaria, discorre sobre o alcance da comunicação. Recomenda-se a utilização no Serviço de Radioamador das frequências de VHF e UHF para coberturas curtas e, de HF, para as longas. Na Seção 7.2, orienta-se que o binômio,

frequência e propagação é fundamental para o equacionamento de um eficaz processo de comunicação, onde considerações sobre distância de comunicação destaca como fator importante na escolha de frequências, equipamentos de rádio e antenas (BRASIL, 2009). A Tabela 4 apresenta um resumo da distância de alcance de propagação, considerando condições de visada direta entre transmissor e receptor

Tabela 4 – Alcance de propagação por faixas.

Alcance Pequeno (0-100 Km)	
$\lambda = 2$ m	144,000-148,000 MHz
$\lambda = 70$ cm	430,000-440,000 MHz
Alcance Médio (0-500 Km)	
$\lambda = 40$ m	7,000-7,300 KHz
$\lambda = 20$ m	14,000-14,350 KHz
$\lambda = 15$ m	21,000-21,450 KHz
$\lambda = 10$ m	28,000-29,700 KHz
Fonte Própria.	

Comparando as Tabelas 2 e 4, nota-se que a faixa de alcance entre 0 a 100 Km, se encontra dentro do espectro de VHF e UHF respectivamente. Segundo Brasil (2009), a faixa de 144-148 MHz, no espectro VHF, é a melhor escolha para comunicação local entre transceptores portáteis (HT, do Inglês *Hand-Talk*) em um raio de aproximadamente 10 Km, com sistema irradiante omnidirecional e, até 30 Km, com antenas direcionais. Ainda de acordo com o autor, a faixa de 430-440 MHz, no espectro UHF, cobre alcances menores, mas possui características semelhantes, inclusive com a possibilidade para o uso de estações repetidoras.

Ainda em análise a Tabela 4, as faixas de alcance médio, entre 100 a 500 Km, estão dentro do espectro HF e, de acordo com Brasil (2009), a faixa de 3500-4000 KHz se apresenta excelente para comunicações noturnas, mas está sujeita a interferências por ruído atmosférico. A faixa de 7000-7300 KHz se mostra excelente para transmissões diurnas e noturnas durante os períodos de baixa atividade solar e, a preferência pelo seu uso deve estar em frequências mais baixas dentro do range do espectro. A faixa de 14000-14350 KHz é uma ótima escolha para distâncias longas independente do horário. Os ranges de frequências de 21000-21450 KHz e 28000-29700 KHz, podem ser utilizadas durante o dia considerando uma alta atividade solar, a última esta sujeita a grandes variações de propagação, mas quando otimizadas, propiciam contatos de alta fidelidade entre o Norte-Nordeste com o Sul/Sudeste. A portaria em sua subseção 7.2.3, ressalta que com uma propagação ideal, qualquer das faixas citadas pode ser utilizada em longas distâncias.

Ao selecionar um subsistema de comunicações para um CubeSat, a escolha do transceptor mais adequado é uma decisão que parte da faixa do espectro de frequência

e do nível de potência no qual se deseja operar, logo existem três possibilidades, que devem ser consideradas: adquirir um transceptor COTS, comprar um projetado para uso terrestre e modificá-lo, ou construir um transceptor a partir de componentes individuais (KLOFAS et al., 2008), para isto é de suma importância observar as características básicas da emissão, as limitações específicas de potência, os planos de faixas com aplicações e demais especificações técnicas complementares estabelecidas por Atos da Superintendência responsável pela administração do uso do espectro de radiofrequências (ANATEL, 2018c).

No contexto deste estudo está a adoção de um transceptor COTS, que trabalhe dentro do espectro de UHF de 430-440 MHz, com potência de transmissão limitada a 1W, vinculado ao projeto e compatível com os requisitos exigidos de baixo custo, dimensões otimizadas e baixo consumo de energia, além dos demais equipamentos de prateleira que compõem a arquitetura do sistema como microcontroladores, antenas, baterias, reguladores de tensão, entre outros, buscando atender as exigências legais de operação impostas pela legislação brasileira regulamentada pela ANATEL.

2.4 Resumo do capítulo

Este capítulo se preocupou em mapear, através de uma revisão bibliográfica, conceitos, características e especificações técnicas que permeiam as condições de operação dos subsistemas de comunicação para satélites, com ênfase em um modelo particular denominado CubeSat. O capítulo ainda destacou os elementos que envolvem um sistema de comunicação via satélite, modelo de propagação, aspectos de transmissão que envolvem faixas de frequências regulamentadas e potência permitida dentro do Serviço de Radioamador por satélite no Brasil.

3 Processo de Desenvolvimento do Sistema

Os avanços da tecnologia viabilizaram as missões espaciais de baixo custo, em plataformas classificadas como satélites de pequeno porte (TOORIAN ARMEN; DIAZ, 2008). Inserido nesta classe, está o padrão CubeSat. O termo é usado para designar um satélite de pequeno porte, em formato cúbico, cujas arestas medem dez centímetros e obedecem a um padrão internacional em relação a estrutura, volume e massa, o qual é descrito por uma especificação de domínio público (PIGNATELLI; MEHRPARVAR, 2013).

Neste cenário, iniciou-se, em 2013, no Laboratório de Simulação e Controle de Sistemas Aeroespaciais (LODESTAR) da Universidade de Brasília (UnB), hoje parte do Laboratório de Ciência e Inovação Aeroespacial (LAICA), o projeto LAICAnSat (BORGES et al., 2018), que tem como principal objetivo a construção de uma plataforma estratosférica, seguindo o padrão CubeSat. A equipe de projeto envolve alunos de graduação e pós-graduação de diferentes áreas de conhecimento. No padrão CubeSat, a abordagem de projeto modular é introduzida por meio do conceito de “containerização”. Contudo, esta abordagem tem apresentado um maior nível de risco de desempenho nos lançamentos, uma vez que a plataforma enfrenta condições extremas de pressão, temperatura, umidade entre outros fatores do meio.

O desenvolvimento integrado de produtos (PUGH, 1991); (ROZENFELD et al., 2006); (BACK et al., 2008) e sistemas (KOSSIAKOFF et al., 2003); (NASA, 2007); (PRIES; QUIGLEY, 2008) para melhorar o desempenho está longe de ser novidade. Contudo, os desafios e paradigmas de integração e desempenho ainda permanecem (HEHENBERGER et al., 2010); (BARBIERI et al., 2014); (BHISE, 2013); (ZHENG et al., 2014); (KILGER et al., 2015).

Este capítulo tem como objetivo apresentar a metodologia selecionada para o desenvolvimento do subsistema de Telemetria e Comando (T&C) de uma plataforma estratosférica, padrão CubeSat. Entende-se por plataforma toda a parte que envolve a estação de comando terrestre e a estação de telemetria embarcada no satélite.

O capítulo encontra-se dividido em três etapas: a primeira etapa, apresenta a revisão da literatura com relação as abordagens para o desenvolvimento de sistemas; a segunda etapa apresenta a proposta de integração do MBD e Modelo V e por fim são realizadas as considerações finais.

3.1 Metodologia

Uma revisão de literatura é um processo de coleta de material descritivo, seleção de categorias e avaliação de materiais, levando à identificação de padrões, temas e questões dentro da literatura (ROZENFELD et al., 2006). A revisão da literatura apresentada neste estudo foi conduzida em duas etapas distintas. A primeira fase, envolveu uma busca de Scopus, EBSCOHost e Google Acadêmico usando combinações dos termos de pesquisa: *Aerospace Design*, *Mechatronic Design*, *Concurrent Engineering*, *System*, e a combinação entre eles. Durante esta fase de coleta de material foram identificados 88 artigos acadêmicos.

Na segunda etapa da revisão da literatura, foram selecionadas abordagens inseridas no contexto da modelagem de sistemas, com ênfase na integração das diferentes disciplinas envolvidas, que pudessem ser replicadas para o contexto do projeto LAICAnSat.

3.1.1 Abordagens para o desenvolvimento de sistemas

Por meio da revisão da literatura foram identificadas sete abordagens para o desenvolvimento do sistema: (HEHENBERGER et al., 2011), (WU et al., 2009), (MHENNI et al., 2014), (ZHENG et al., 2016), (HEHENBERGER; ZEMAN, 2007), (GAUSEMEIER; MOEHRINGER, 2002), (PIETRUSEWICZ, 2019), sintetizadas na Tabela 5.

Tabela 5 – Modelos de Referência.

AUTOR	ABORDAGEM
(HEHENBERGER et al., 2011)	Apresenta uma visão geral sobre os modelos para o desenvolvimento de produtos usados no projeto mecatrônico, analisa o fluxo de informações através de ferramentas e considera a consistência do modelo.
(WU et al., 2009)	Propõe uma estrutura de decomposição funcional orientada a objetos, desenvolvida para co-design e co-análise de funções de produto, estruturas e suas relações de mapeamento de uma maneira <i>top-down</i> . A estrutura possui três modelos operacionais: Modelo de Função, Modelo de Objeto e Modelo de Fluxo de Informações.
(MHENNI et al., 2014)	Proposta de uma metodologia baseada em SysML. Essa metodologia consiste em duas fases: uma análise de caixa preta com um ponto de vista externo que fornece um conjunto abrangente e consistente de requisitos e uma análise de caixa branca que conduz progressivamente à arquitetura interna e ao comportamento do sistema.
(ZHENG et al., 2016)	Apresenta um modelo de interface multidisciplinar para o projeto de sistemas, afim de permitir a integração entre os membros da equipe de design formada pelas diferentes disciplinas.
(HEHENBERGER; ZEMAN, 2007)	Apresenta o modelo hierárquico, com objetivo de integrar as disciplinas envolvidas, desde o início do desenvolvimento, buscando auxiliar na organização dos modelos propostos por meio da separação das representações em conhecimento estrutural, comportamental ou funcional.
(GAUSEMEIER; MOEHRINGER, 2002)	Apresenta o modelo V, adaptado da engenharia de <i>software</i> para a realidade do desenvolvimento de projeto de sistemas, propõe uma interação entre as diferentes equipes de desenvolvimento, onde o sistema total, as subfunções e subsistemas são desenvolvidos em cooperação e simultaneamente pelas equipes envolvidas.
(PIETRUSEWICZ, 2019)	Propõe uma abordagem baseada no modelo <i>Model-Based Design</i> (MBD), como exemplo para projetar novos sistemas de controle, combinando ferramentas de modelagem, simulação computacional e geração do código.

Fonte Própria.

3.1.1.1 Compreensão do desenvolvimento do produto

Seguindo a proposta de equilíbrio entre as diferentes disciplinas envolvidas, Hehenberger et al. (2011), apresentam, conforme ilustrado na Figura 11 aspectos que permitem análises de diferentes ângulos em relação ao processo do projeto de sistemas, com pontos de vistas que incluem os seguintes atributos:

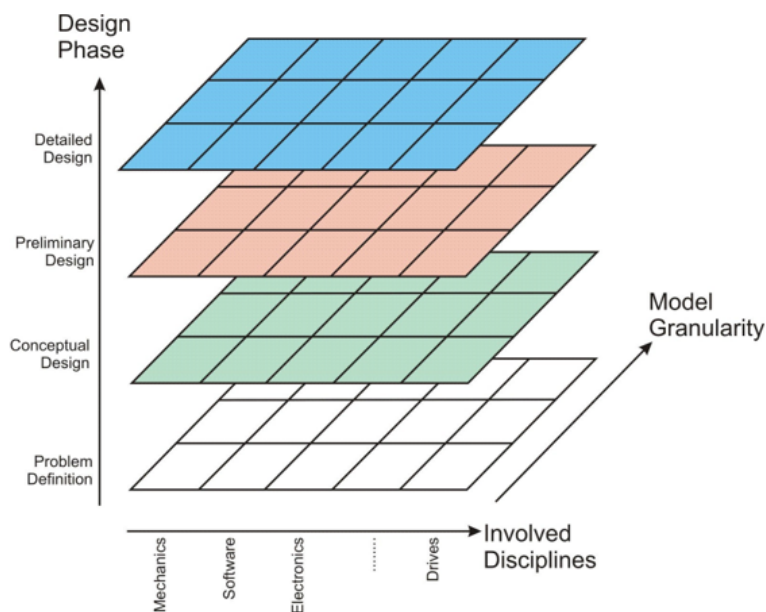


Figura 11 – Ponto de vista do modelo do produto.
Adaptado de: Hehenberger et al. (2011).

- Objetos e modelos de diferentes disciplinas, que estão envolvidos no processo de desenvolvimento de um sistema multidisciplinar;
- Granularidade do modelo, denotando a extensão em que um objeto ou modelo é dividido em partes menores;
- Fases de projeto: O processo de desenvolvimento pode ser estruturado em quatro fases, ou seja, definição de problemas, projeto conceitual, projeto preliminar e projeto detalhado.

Ao projetar um sistema que envolve uma equipe multidisciplinar, formada por mecânicos, eletrônicos e desenvolvedores de sistemas, é possível projetar um componente do equipamento, por exemplo mecânico, antes do início de qualquer projeto de sistema de controle. Uma desvantagem é a possível falta de compatibilidade entre os subsistemas, que resulta em esforços e custos adicionais para atender as especificações do sistema total. Hehenberger et al. (2011), reiteram que outra desvantagem dessa abordagem é que, durante o processo de projeto, é necessário tomar decisões, onde os engenheiros de projeto precisam equilibrar soluções mecânicas, eletrônicas e de *software*.

3.1.1.2 Estrutura de decomposição orientada a objetos

Com foco na integração da equipe, no trabalho de Wu et al. (2009), os autores buscam atender a dois propósitos. O primeiro é ser uma ferramenta de coordenação e comunicação para a equipe multidisciplinar por meio da colaboração, análise, e síntese do projeto no nível do produto, composto de peças eletrônicas, de *software* e mecânicas.

O segundo é ser um ponto entre os projetistas e os engenheiros de desenvolvimento em especificações de projeto e atribuições de tarefas.

A Figura 12, apresenta a estrutura de blocos e de etapas, onde o modelo de funções é representado por uma FT (do Inglês, *Function Trees*), o modelo de objeto é representado por uma HOOM (do Inglês, *High Order Object Model*) e o modelo de fluxo de informações é representado por um FOMM (do Inglês, *Function & Object Mapping Model*), formando a técnica hierárquica e modelagem funcional OO (do Inglês, *Object-oriented*).

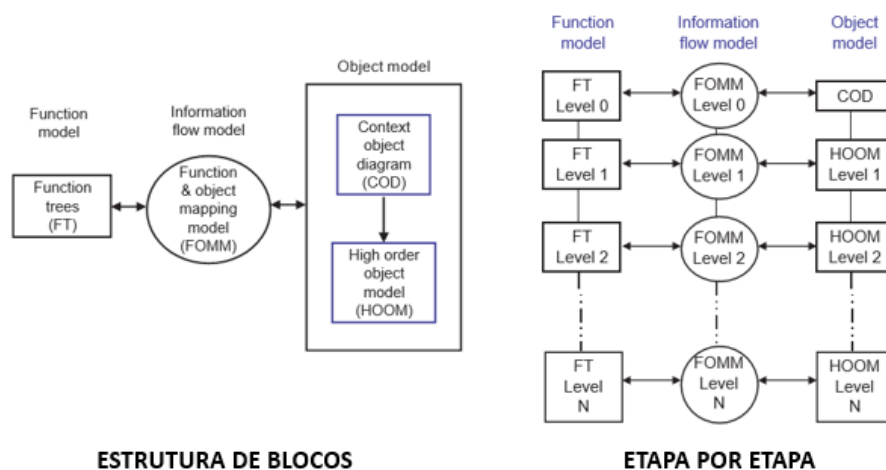


Figura 12 – Técnica hierárquica de modelagem OO.
Adaptado de: Wu et al. (2009).

Wu et al. (2009), conceitua o HOOM como sendo uma ferramenta de decomposição de objetos com análise de estrutura de alta ordem para objetos primitivos. Estes objetos podem ser facilmente compreendidos pelos projetistas e não requerem análise adicional. Objetos de alta ordem, no entanto, precisam ser decompostos. Um objeto possui atributos e métodos. Esses métodos podem ser executados por seu objeto para atender aos requisitos no domínio funcional.

Os autores ainda acrescentam a técnica, o modelo de fluxo de informações, FOMM, como um modo de análise e verificação para descrever as relações de mapeamento e parâmetros de implementação entre modelos de função e objeto. A verificação das relações de mapeamento refere-se a quais objetos no modelo de objeto cumprem as funções do modelo de função. A análise dos parâmetros de implementação, por outro lado, refere-se a quais fluxos de informação são necessários para os objetos implementarem as funções mapeadas, considerando os três tipos de fluxos da modelagem funcional tradicional: material, energia e sinal.

3.1.1.3 Metodologia baseada em SysML

O SysML foi projetado para fornecer construções simples, mas poderosas, para modelar uma ampla gama de problemas de engenharia de sistemas. É particularmente eficaz na especificação de requisitos, estrutura, comportamento, alocações e restrições nas propriedades do sistema para suportar a análise de engenharia, conceitua Mhenni et al. (2014).

Mhenni et al. (2014) propõe como metodologia uma abordagem *top-down*. Iniciando com uma análise de caixa fechada onde o sistema de interesse é visto como uma caixa preta, o que significa que sua estrutura interna ainda não é definida. Essa fase de modelagem é um ponto de vista externo do sistema que visa definir um conjunto consistente de requisitos que o sistema deve satisfazer e que será a linha de base para as próximas etapas, conforme ilustra a Figura 13.



Figura 13 – Abordagem *Top-down*.
Fonte Própria.

Em seguida, faz-se uma análise interna da caixa, onde a mesma é aberta com o intuito de modelar sua estrutura interna e seu comportamento, com relação ao conjunto de requisitos especificados durante a análise caixa-preta. Uma linha de base está agora disponível para seguir adiante e identificar diferentes soluções candidatas.

3.1.1.4 Modelo de Interface Multidisciplinar

O projeto de sistemas requer uma colaboração multidisciplinar que frequentemente leva à iteração durante o processo de projeto simultâneo. Portanto, o gerenciamento da compatibilidade de interfaces é muito importante para este tipo de sistema, pois pode ajudar os projetistas a garantirem consistência entre diferentes equipes de projeto e evitar erros de projeto durante o processo de engenharia colaborativa, o que pode reduzir bastante as iterações desnecessárias, conforme sugere Bettig e Gershenson (2010).

Zheng et al. (2016) destaca que as necessidades do modelo de interface multidisciplinar para o projeto de sistemas, são definidas como, “*Ports*”, que representam os principais locais através do qual uma parte do sistema interage com outras partes do ambiente, conforme ilustra a Figura 14. As portas, em tradução para o Português, são representadas pelas seguintes entidades:

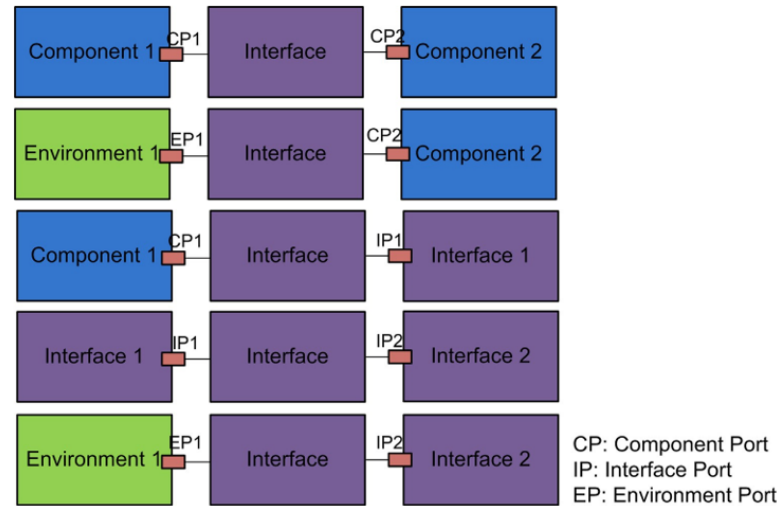


Figura 14 – Tipos de *ports*.
Fonte: Zheng et al. (2016).

- CP (do Inglês, *Component Port*) – Ponto de conexão de um atributo que interage com outros componentes de um sistema;
- IP (do Inglês, *Interface Port*) – Local que afeta outros componentes do sistema;
- EP (do Inglês, *Environment Port*): Simboliza a interferência de fatores externos do ambiente em relação a componentes do sistema.

Do ponto de vista de Zheng et al. (2016), a colaboração multidisciplinar torna cada vez mais informações confidenciais (por exemplo, dados financeiros ou de clientes). A visibilidade do atributo da classe *Port*, com seus níveis de segurança "*public*", "*protected*" e "*private*", é usada para descrever como o parâmetro e o documento vinculados a uma porta podem ser acessados, garantindo maior controle e segurança.

3.1.1.5 Projeto Hierárquico

O modelo hierárquico é um tipo de método de desenvolvimento de sistemas, proposto por Hehenberger e Zeman (2007), que emprega técnicas envolvendo uma rigorosa integração entre aspectos mecânicos, elétricos, eletrônicos, de controle e de *software*.

Segundo Mlambo et al. (2018), o modelo de design hierárquico considera a integração das disciplinas desde o início do desenvolvimento. Este método auxilia na organização

dos modelos propostos separando as representações em conhecimento estrutural, comportamental ou funcional. Neste contexto, o sistema é dividido em módulos, os quais são subdivididos em subsistemas específicos e integrados a disciplina correspondente, permitindo diferentes graus de detalhamento e pontos de vista. A Figura 15 ilustra um exemplo referente a um módulo formado por componentes de domínio específicos.

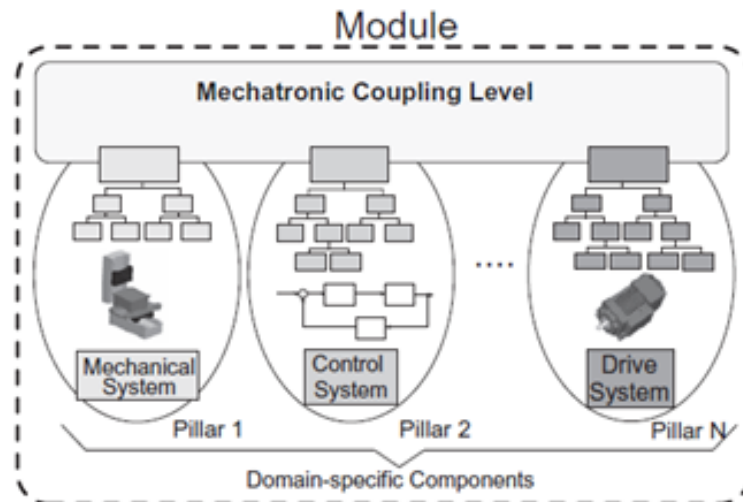


Figura 15 – Módulo unificado.
Fonte: Hehenberger et al. (2010).

A descrição do modelo da Figura 15 é estruturada da seguinte forma: o nível superior representa o acoplamento realizando uma interface com os outros pilares, neste ponto observa-se a junção entre os pilares do modelo. Os componentes do domínio específico representam a modelagem detalhada de cada subsistema em particular, com as suas propriedades e comportamentos individuais, permitindo refletir a natureza das atividades envolvidas em cada nível individual dentro do sistema como um todo.

A separação das interfaces em componentes, permite a inserção de outras interfaces e devido a sua característica modular, viabiliza o gerenciamento de um grande número de interfaces, ampliando o poder de conhecimento de projeto, gerenciamento de complexidade, atualização, evolução, trabalho em paralelo das equipes e substituição de partes do sistema (MLAMBO et al., 2018). A capacidade de decompor uma tarefa de projeto fornece a base para alcançar soluções criativas de design (KOMOTO; TOMIYAMA, 2012).

3.1.1.6 Modelo V

O VDI2206 (GAUSEMEIER; MOEHRINGER, 2002), sugere a realização do processo para o desenvolvimento de sistemas, relacionando os domínios específicos, de acordo com o chamado modelo em V ilustrado na Figura 16.

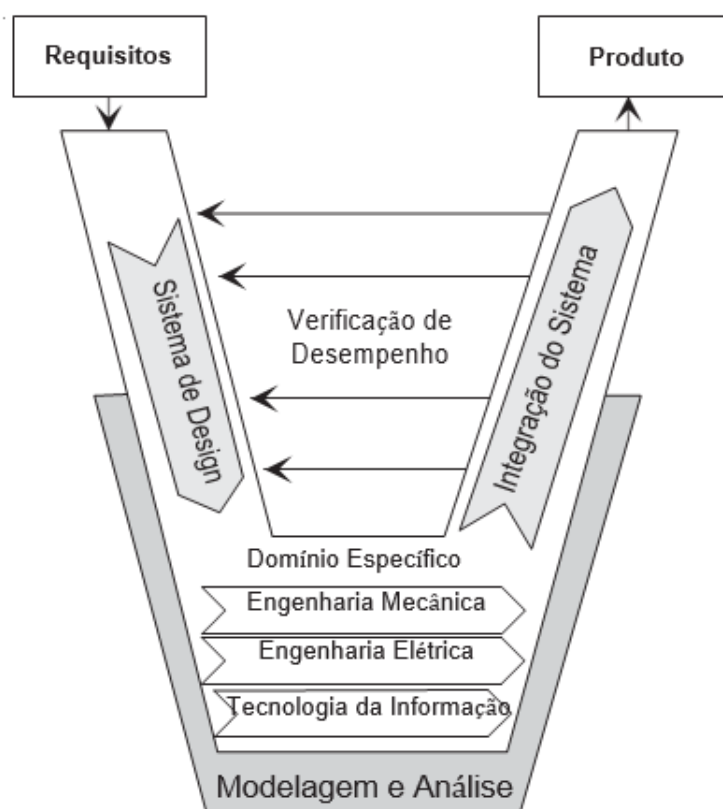


Figura 16 – Modelo V para desenvolvimento de sistemas.
Adaptado de: Hehenberger et al. (2010).

Hehenberger et al. (2010), Figura 16, destacam que após uma análise integral dos requisitos do sistema, se define subfunções e subsistemas, esta etapa se encontra situada no ramo esquerdo do modelo V e são desenvolvidas de forma cooperativa e simultânea pelas equipes envolvidas. Após verificação das subfunções e testes dos subsistemas, estes são integrados gradativamente, representando a ramificação direita do modelo V. Em seguida, o desempenho da integração do sistema será verificado, caso sejam constatadas melhorias, a fase inicial de operação se repete, determinando um processo de interação até a estabilidade e coesão do sistema no ponto de vista da equipe.

O modelo V, devido a sua origem no desenvolvimento de sistemas de *software*, se mostra bastante atrativo na adoção no desenvolvimento de sistemas que envolvem domínios específicos, por este envolver em seu processo elementos bastante importantes que agrupam programação e manipulação de códigos.

3.2 Modelo MBD (*Model-Based Design*)

O *Model-Based Design* (MBD), merece uma seção específica por se tratar do modelo de referência adotado neste estudo, sua escolha em relação às demais apresentadas, se dá, pela praticidade no desenvolvimento de modelos, durante a etapa de integração das

diferentes áreas envolvidas, proporcionada pela redução das etapas de não desenvolvimento, combinando as etapas de projeto, implementação e teste em um único processo, ou seja, os modelos são criados e testados concomitantemente às etapas de produção, permitindo aos envolvidos no processo maior controle e realização de testes simultâneos, resultando em maior interação e confiança entre a simulação e a prototipagem. Conceitos, procedimentos, componentes e *softwares* utilizados neste método serão apresentados, buscando neste primeiro momento, identificar o que fazer no contexto do problema da pesquisa.

3.2.1 Modelo tradicional versus MBD

Barbalho (2006) destaca, que em projetos multidisciplinares, geralmente há uma área dominante em termos de tecnologia de produto, a qual direciona o fluxo principal de projeto da empresa. A Figura 17 apresenta um quadro com as etapas genéricas do projeto mecânico, eletrônico e *software*, proposto por Bernardi et al. (2002). Estas etapas e a integração das áreas exemplifica a necessidade de adoção do MBD para o caso deste estudo.

Bernardi et al. (2002) consideram haver uma forte diferenciação entre os projetos eletrônico e de *software* se comparados com o projeto mecânico quanto à questão dos testes necessários ao produto. A mecânica trata de uma área mais exata, com cálculos mais precisos e simulações por elementos finitos para definir margens de segurança para o produto. Em *software* e eletrônica, procedimentos de testes com protótipos físicos são normalmente necessários uma vez que há problemas relacionados com o nível de ruído ou carregamento térmico dos circuitos eletrônicos que dificilmente se consegue detectar por simulação, além de falhas de codificação humanas, na programação embarcada, que ocasionam em problemas de desempenho e eficiência da eletrônica.



Figura 17 – Fluxo principal de atividade de projeto mecânico, eletrônico e de *software*.
Fonte: Bernardi et al. (2002).

Diferenças importantes dentro do fluxo tradicional do desenvolvimento de produto podem ser identificadas em projetos mecânicos, eletrônicos e de *software*, sendo destacadas por Buur (1990) e apresentadas abaixo de forma resumida:

- em relação às funções principais: em eletrônica e *software* a função principal é a transformação de informações, através de sinais elétricos (eletrônica) ou funções lógicas (*software*), enquanto que em engenharia mecânica é possível transformar materiais e energia, além de informações;
- em relação ao projeto conceitual: em mecânica, os problemas podem ser sempre caracterizados como novos havendo muitas alternativas de solução disponíveis. Já em eletrônica utiliza-se componentes padronizados (COTS) e soluções de projeto sugeridas por fabricantes. Em *software* há um conjunto pré-estabelecido de operações pelas quais o projeto poder ser implementado, sendo a priori, incomum a possibilidade de usar algoritmos padrão;
- em relação ao projeto detalhado: há uma grande quantidade de tecnologias de manufatura mecânica havendo alto grau de liberdade para o projetista especificar dimensões, formas, materiais e acabamentos. Em eletrônica há uma quantidade menor de tecnologias de produção sendo que os componentes são normalmente comprados. Em *software* não há uma fase de produção propriamente dita, uma vez que a própria codificação gera o produto final.

Segundo Pietrusewicz (2019), o investimento no desenvolvimento baseado em modelo é justificado nos seguintes casos: o sistema modelado é grande, complexo ou de domínio cruzado por natureza, por exemplo, contém componentes mecânicos, eletrônicos, *software*, como apresentado na Figura 17 e explicado no parágrafo anterior, além disso o sistema modelado é miniaturizado, a prova de experimentos conceituais são muito demorados ou caros e, finalmente, se o sistema desenvolvido for modificado no futuro ou simplesmente não estiver disponível ou ainda não existir no momento do desenvolvimento.

A grande desvantagem do método de desenvolvimento de produtos tradicional está interligada ao custo e tempo necessário para a construção de protótipos em cada iteração de avaliação dos resultados. O ciclo de construir e testar torna difícil a previsão de quanto tempo levará para que todo o processo de desenvolvimento seja concluído. Além disso, longas iterações exigem bastante trabalho, tornando impossível explorar todas as configurações de desenvolvimento, fazendo com que haja uma grande chance que o projeto final não seja otimizado (MALONEY; NURSILO, 2011). No MBD, engenheiros usam simulação para refinar as especificações antes de construir o protótipo de teste físico.

Os seguintes benefícios podem ser atribuídos à abordagem do MBD, conforme cita Pietrusewicz (2019):

1. Foco na modelagem virtual;
2. Conformidade do produto final com os requisitos especificados;
3. Correção de erros em tempo real;
4. Redução de erros de *software*;
5. Verificação dos objetivos do projeto sem construir um protótipo físico;
6. Desenvolvimento de cenários de casos de teste e sua validação em um ambiente de simulação;
7. Gerenciamento mais fácil de projetos complexos.

3.2.2 Matriz de Adoção

Aarenstrup (2015) apresenta o modelo de nove caixas de imersão no processo de desenvolvimento de algoritmos de controle baseado no Modelo de Projeto (Figura 18). Esta representação em caixas é também conhecida como Matriz de Adoção, onde empresas e grupos de desenvolvimento com maior grau de imersão, Figura 18, caixa MBD-9, empregam uma abordagem totalmente baseada em modelos durante a fase de pesquisa e desenvolvimento. O mínimo, no entanto, aplica uma representação gráfica dos componentes do sistema de controle desenvolvido, Figura 18, caixa MBD-1.

Pietruszewicz (2019) aponta duas áreas importantes no desenvolvimento de sistemas representado pelos fluxos contidos no eixos x e y, sendo:

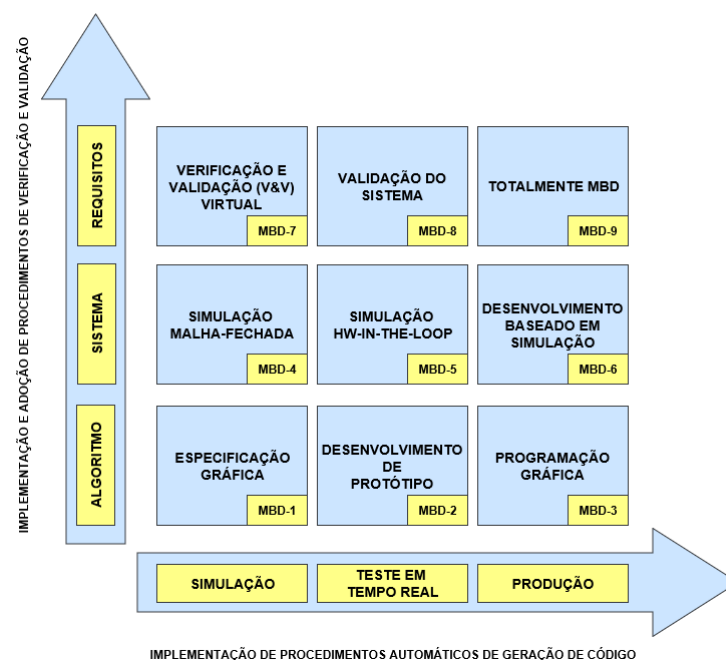


Figura 18 – Matriz de Adoção.
Adaptado de: Opencadd (2019) e Pietruszewicz (2019).

1. Eixo y - Implementação e adoção de procedimentos de verificação e validação:

- (i) Modelagem de algoritmo de controle (malha aberta) - sem suporte de validação; as especificações incluem apenas informações de nível operacional;
- (ii) Simulação de projeto do sistema (malha fechada) - são usadas para fornecer informações para projetistas; o teste em tempo real antes da transição do sistema de controle, para uma plataforma de destino, permite eliminar a maioria dos erros do produto final, sem o risco de danificar seus componentes físicos (por exemplo, sensores, atuadores);
- (iii) Desenvolvimento baseado em requisitos - o gerenciamento de requisitos está consolidado no processo de desenvolvimento do sistema.

2. Eixo x - Implementação de procedimentos automáticos de geração de código:

- (i) Simulação - a geração de código suporta apenas cálculos de simulação dinâmica e do sistema; nenhum código em tempo real é gerado;
- (ii) Testes em tempo real - os testes em tempo real são suportados por procedimentos automáticos de geração de código;
- (iii) Código de produção - esse nível mais alto de desenvolvimento de *software* é obtido pela utilização de aplicativos que suportam a geração de código para o produto final diretamente das especificações.

As caixas de imersão apresentadas na Matriz de Adoção são explicadas na Tabela 6 para o devido entendimento das etapas que regem o fluxo da Figura 18:

Tabela 6 – Caixas de imersão da Matriz de Adoção.

NÍVEL	TIPO	DESCRIÇÃO
MBD-1	Especificação Gráfica	Especificações de <i>software</i> do sistema projetados graficamente, através de uma linguagem de modelagem, para uso em outros aplicativos de <i>software</i> (<i>AutoCAD</i> , <i>Simulink</i> [®]) ou como artefatos semânticos (código-fonte, documentação textual, esquemas de sistemas de controle).
MBD-2	Desenvolvimento de Protótipo	Procedimentos de geração do código de destino, com base nas especificações gráficas do modelo de <i>software</i> , obtidas em MBD-1 para verificar o comportamento do sistema no ambiente de destino.
MBD-3	Programação gráfica	Nesse nível, o código do sistema de controle é gerado inteiramente por meio de especificações gráficas. Os arquivos binários são o resultado de um processo de desenvolvimento de <i>software</i> totalmente automatizado, que leva em consideração o tipo de sistema de destino e seu ambiente de trabalho.
MBD-4	Simulação Malha-Fechada	Modelos do objeto controlado (ambiente) e <i>software</i> de controle, simulam em conjunto dentro de um circuito de controle fechado, um sistema que inclui sinais de entrada do operador e sinais de resposta de sensores, cuja operação também é simulada.
MBD-5	Simulação <i>hardware</i> em <i>loop</i>	Um tipo de simulação em que parte do <i>software</i> de controle, o modelo do objeto de controle e o ambiente no qual o objeto de controle ocorre, estão sujeitos à verificação em tempo real. Embora o código de controle seja executado em um dispositivo físico, não há perigo de danificar os componentes físicos do sistema, pois todos os sinais de saída e entrada são simulados.
MBD-6	Desenvolvimento Baseado em Simulação	O código de resultado do sistema de controle e o <i>hardware</i> usado na aplicação, são testados levando em consideração o modelo do objeto e o modelo do ambiente no qual o sistema é usado. A plataforma de destino e seus algoritmos, são validados como teste de simulação e viabilidade computacional em regime de tempo real. Esses testes são essenciais para garantir a qualidade da solução e sua integridade a falhas.
MBD-7	Verificação e validação virtual	Requisitos para o sistema projetado são testados e verificados por meio de simulações de computador. A principal diferença entre este nível e o diretamente abaixo dele, o MBD-4, é que os requisitos do projeto são fatorados no processo de design.
MBD-8	Validação do sistema	Casos de testes utilizados em simulações computacionais são aplicados para execução de novos testes em objetos reais. Requisitos para o sistema projetado são testados e verificados por meio de simulações de computador.
MBD-9	Implementação completa do MBD	O último nível de imersão integra todos os estágios acima mencionados, em um processo de projeto totalmente automatizado.

Fonte Própria.

3.2.3 Proposta de Integração das Metodologias: MBD e Modelo V

Por meio da revisão bibliográfica com intuito de selecionar abordagens que pudessem ser replicadas no contexto do projeto LAICAnSat, observou-se que a compreensão do MBD, pode-se relacionar ao modelo V, por meio da representação gráfica do ciclo de desenvolvimento do produto e das principais etapas em conjunto com as entregas entre o ambiente de desenvolvimento, auxiliando na resposta a questões sobre o processo de pesquisa e desenvolvimento.

O diagrama V da Figura 19 ilustra o processo de desenvolvimento sobreposto à Matriz de Adoção do MBD, neste cenário pode-se modelar e simular uma aplicação produto a partir de modelos teóricos, extraídos das caixas de imersão e estimar parâmetros e potenciais falhas em simulação obtendo a melhor solução com maior rapidez.

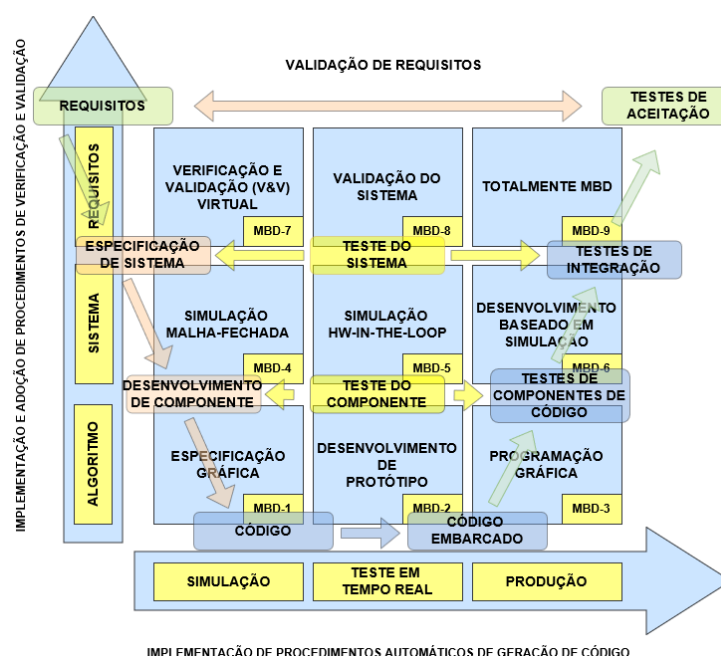


Figura 19 – Matriz de Adoção integrada ao Modelo V. Adaptado de: Opencadd (2019) e Hehenberger et al. (2010).

De acordo com Stella et al. (2015), o MBD se inicia com o mesmo conjunto de requisitos do processo tradicional. Porém, ao invés de servir para desenvolver especificações de modo textual, os requisitos são utilizados para desenvolver um modelo executável. Utilizando ferramentas computacionais é possível simular o sistema, descobrindo falhas e defeitos, ou seja, antecipar possíveis problemas em relação a implementação. Com o modelo finalizado e verificado, é possível gerar automaticamente o código e refazer testes a partir destes códigos, como o desenvolvimento é realizado de forma integrada, tudo é realizado dentro do mesmo ambiente de desenvolvimento, facilitando testes já em modelos iniciais e validando se os requisitos estão sendo alcançados.

A Figura 20 apresenta o fluxo de trabalho simplificado do MBD. As etapas se

relacionam, auxiliando os profissionais envolvidos no tratamento de falhas independente da fase de implementação, observado pela inversão no fluxo entre a elaboração do modelo e a verificação contínua, ou seja, esta rotina de verificação é expressa pelos testes de componentes, de sistemas e de validação de requisitos presentes no Modelo V, denominado Verificação de Componentes apresentado na Figura 16.

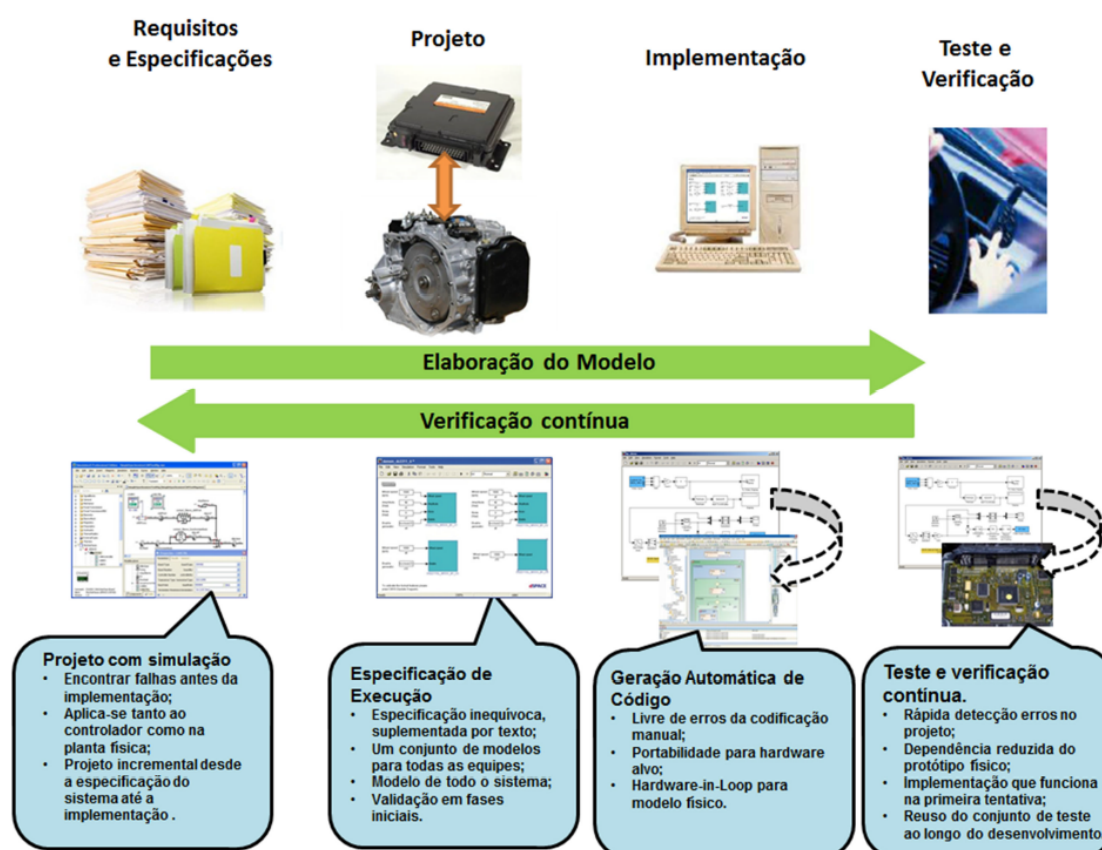


Figura 20 – MBD simplificado.
Fonte: Stella et al. (2015).

No MBD proposto para plataformas que envolvem elementos mecânicos, eletrônicos e de *software*, o desenvolvimento segue basicamente as mesmas etapas apresentadas no diagrama do Modelo V, conforme lista a seguir:

- (a) Definição dos requisitos;
- (b) Especificação de sistema (Projeto da arquitetura funcional);
- (c) Desenvolvimento da arquitetura física;
- (d) Testes de componente (Unidade);
- (e) Testes de integração;

- (f) Verificação e validação.

Em resumo, define-se os requisitos importantes para o funcionamento do sistema. A partir da sua descrição textual é possível a definição da especificação do sistema, resultando em sua arquitetura. Com o protótipo finalizado aplica-se o modelo funcional para gerar o código na linguagem desejada que será embarcado em um *hardware* escolhido. Com o modelo desenvolvido em *software* e o código gerado é possível verificar o funcionamento do projeto utilizando entradas de teste já desenvolvidas. Com todas as funções verificadas, pode-se embarcar o código no sistema final e validá-lo como um todo.

Tomando como diretrizes os parâmetros e etapas do MBD, com a integração de métodos do Modelo V, o qual auxilia o trabalho entregando as ferramentas necessárias conforme o avanço da pesquisa, tal qual, exposto anteriormente, busca-se o atendimento da modelagem da interface de comunicação ponto-a-ponto do tipo *half-duplex* dedicado a missões espaciais, aplicado a plataformas estratosféricas. Logo a modelagem proposta assume o seguinte fluxo em atendimento as etapas do Modelo V:

- (a) Definição dos requisitos;
- (b) Construção da arquitetura;
- (c) Modelo Virtual;
- (d) Implementação;
- (e) Testes e verificação;

A Figura 21 apresenta o fluxo adaptado para o Modelo V, partindo da definição dos requisitos e arquitetura, localizada na parte superior esquerda do modelo, seguindo na direção descendente para o modelo virtual. O lado direito, em paralelo realiza os processos de verificação e validação, indo de encontro a aplicação dos métodos de testes para MBD: MIL, SIL, PIL e HIL, que são executados concomitantemente junto com cada etapa do lado esquerdo até o alcance dos resultados esperados relacionados aos requisitos, arquitetura e modelo virtual. Com o atendimento ao escopo do projeto, passa-se a implementação do protótipo.

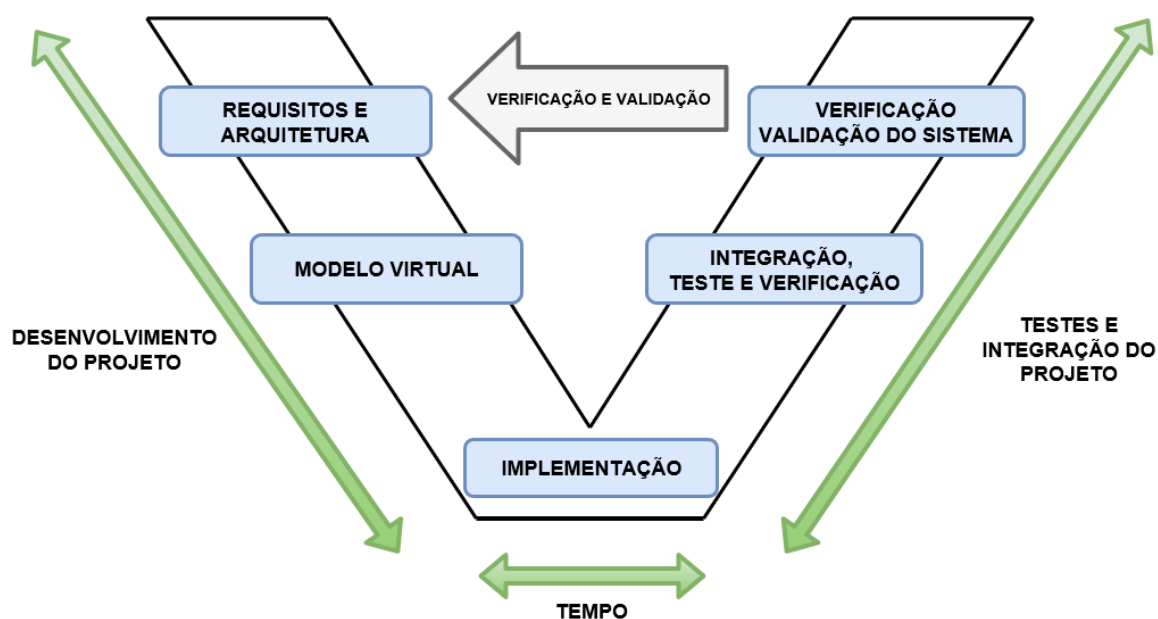


Figura 21 – Aplicação Modelo V.
Fonte Própria.

Ainda em relação ao diagrama apresentado na Figura 21, vale destacar que a parte esquerda diz respeito a fase de desenvolvimento do projeto e o lado direito está vinculado a fase de testes e integração do projeto, a relação das duas, permite uma economia de tempo durante a etapa de implementação, além de minimizar os custos do protótipo físico e permitir maior confiança no processo de construção e funcionalidades.

3.2.4 Técnicas adotadas para testes e verificações

Testes e verificações certificam que o produto atenderá às normas e atuará da forma esperada, sem surpresas ou situações de funcionamento inesperadas, contemplando todas as possíveis circunstâncias que se possa encontrar, dessa forma a qualidade do produto está diretamente vinculada a necessidade de adoção destas técnicas, independente da finalidade do modelo, seja a concepção de um produto final ou apenas para observação de um comportamento, os testes se justificam e são valiosos para confirmação da veracidade da informação coletada, garantia de operação do *hardware*, de acordo com a simulação virtual.

A metodologia sugere os seguintes testes. Estes acompanham o modelo desde a sua criação até os estágios finais, conforme destaca Technology (2018):

1. **Testes Funcionais:** Estes testes estão relacionados com o funcionamento do algoritmo e com as expectativas do desenvolvedor. Eles têm como função validar o algoritmo implementado. Portanto, os testes funcionais dirão, por exemplo, se o *software* de controle está atuando com o tempo de resposta correto, se ele faz com

que o sistema atinja o valor de operação correto, se a resposta do *software* de controle está contemplando todas as possíveis entradas, etc. Estes tipos de testes auxiliam na compreensão da equipe sobre o processo. Muitas vezes, com a diversificação dos testes funcionais, é possível observar cenários que antes não haviam sido considerados, resultando na mudança do algoritmo implementado ou até dos requisitos de desempenho do projeto.

2. **Testes Estruturais:** Estes testes focam na forma em que a lógica do algoritmo foi estruturada. Neste caso, testa-se, se todos os possíveis estados condicionais do algoritmo podem ser acessados, se existem caminhos de lógica ociosos em que o algoritmo nunca acessa em circunstância alguma, ou se existe algum estado irreversível, entre outros.
3. **Testes de Robustez:** Nesta etapa o algoritmo é exposto a situações extremas, afim de observar o seu comportamento, como por exemplo: verificar se em algum momento, alguma variável sofrerá *overflow* de memória, ou se existe circunstâncias em que ocorrerão divisões por zero, e ainda analisar até que ponto o sistema conseguirá manter a estabilidade da planta.
4. **Testes de *Compliance*:** Relacionado a adequação do algoritmo, segundo normas e padrões de qualidade. Ferramentas como o *MathWorks* que será abordada ainda neste capítulo, disponibilizam estes testes já no ambiente de desenvolvimento.

No MBD, as técnicas tradicionalmente utilizadas para validação são: *Model-In-the-Loop* (MIL), *Software-In-the-Loop* (SIL), *Processor-In-the-Loop* (PIL) e *Hardware-In-the-Loop* (HIL) (STELLA et al., 2015). Geralmente a aplicação destas técnicas seguem a sequência cronológica apresentada na Figura 22.



Figura 22 – Sequência de técnicas para verificação de modelos
Fonte Própria.

3.2.5 Ferramentas de apoio a modelagem

As técnicas para verificação de modelos, contam como o apoio de ferramentas de *software* para modelagem e engenharia de sistemas disponíveis no mercado. Os mais populares estão listados na Tabela 7. Essas ferramentas são as mais adotadas devido a seus recursos avançados e suporte para desenvolvimento orientado para equipes.

A arquitetura de *software* de modelagem e a pesquisa de simulação de novos produtos, incluindo sistemas de controle, são possíveis devido aos recursos presentes nas aplicações citadas na tabela 8.

Tabela 7 – Ferramentas de *software* para modelagem em equipe

SOFTWARE	FABRICANTE	SITE
Came Systems Modeler	NoMagic	< http://www.nomagic.com/products/cameo-systems-modeler.html >
Innoslate	SPEC Innovations	< https://www.innoslate.com >
Enterprise Architect	Sparx Systems	< http://www.sparxsystems.com/products/mdg_sysml.html >
Modelio SA	ModelioSoft	< http://www.modeliosoft.com/en/products/solutions/system-architect-solution-overview.html >
Eclipse Papyrus	Laboratório de Engenharia Dirigida por Modelos para Sistemas Embarcados (Comissão de Energias Alternativas e Energia Atômica da França)	< http://www.eclipse.org/modeling/mdt/papyrus >
Design Rhapsody Developer	IBM	< http://www.ibm.com/software/rational/products/rhapsody/sysarchitect >
ARTISAN Studio	PTC	< http://www.atego.com/products/artisan-studio >

Fonte Própria.

Tabela 8 – Ferramentas para arquitetura e simulação de produtos

SOFTWARE	FABRICANTE	CÓDIGO ABERTO	SITE
Matlab/Simulink	Mathworks	-	< https://la.mathworks.com/products/simulink.html?requestedDomain= >
MapleSim	Maplesoft	-	< https://www.maplesoft.com/products/maplesim/ >
Simcenter Amesim	Siemens	-	< https://www.plm.automation.siemens.com/global/pt/products/simcenter/simcenter-amesim.html >
Scilab	Scilab Enterprise	x	< https://www.scilab.org/ >
HOPSAN	Linkoping University	x	< https://liu.se/en/research/hopsan >
OpenModelica	OSMC	x	< https://openmodelica.org/ >
Dymola	Dessault System	-	< https://www.3ds.com/products-services/catia/products/dymola/ >

Fonte Própria.

O *Simulink*[®] pacote integrante do *software Matlab*[®], desenvolvido pela *Mathworks* foi a opção escolhida como ferramenta de suporte adotada neste estudo, os motivos desta preferência se encontra em sua ampla gama de aplicações que envolvem áreas como: automotiva, aeroespacial, semicondutores, comunicações e design de sistemas autônomos. Maiores detalhes sobre estas ferramentas são explanados na subseção a seguir.

3.2.5.1 *Software Matlab/Simulink*[®]

Em grande parte do projeto, principalmente na etapa de desenvolvimento do modelo foram utilizados os *softwares Matlab*[®] e *Simulink*[®], O *Matlab*[®] é uma linguagem utilizado para análise precisa e eficiente de dados, modelos matemáticos, simulação de sistemas e desenvolvimento de produtos. Possui uma ampla biblioteca de funções matemáticas, que permite a criação de novas funções, algoritmos customizados, maior eficiência no tratamento de dados por meio da análise gráfica e ainda possui interface de integração com *hardwares*, possibilitando o desenvolvimento de sistemas com entradas e saídas para outros dispositivos (OPENCADD, 2019).

O *Simulink*[®] é um aplicativo incorporado ao *Matlab*[®], tem como principal objetivo a modelagem matemática de sistemas dinâmicos, permitindo ao usuário a criação de modelos para análise e controle de sistemas que sofrem alterações no decorrer do tempo. Neste cenário, o *Simulink*[®], auxilia na construção de diagrama de blocos que possibilitam

a simulação de condições que se deseja modelar, sendo bastante aplicado em situações reais das mais variadas áreas: aeroespacial e defesa, automotiva, comunicações, eletrônica, mineração, processamento digital de sinais, química, petroquímica entre outros.

Segundo (OPENCADD, 2019), a modelagem e simulação com *Simulink*[®], é uma maneira barata aplicada ao dimensionamento do produto, pois este é realizado de forma interativa, absorvendo durante o seu desenvolvimento, informações importantes sobre a dinâmica do sistema e as influências dos componentes integrados, ou seja, toda a etapa de dimensionamento é realizada por simulações do comportamento do modelo, permitindo cenários de verificações e testes em condições severas de operações que poderiam vir a causar falhas e/ou condições de perigo ao usuário ou equipamento, antecipando a problemas futuros, que podem se refletir no equipamento físico, sendo que todas estas condições são simuladas via *software*, sem a necessidade de construir protótipos, que eleva o custo do projeto.

A ferramenta *Simulink*[®] integrada ao *software Matlab*[®] se mostra bastante eficaz, como ferramenta de auxílio na implementação MBD, modelo de referência adotado neste estudo. A escolha deste conjunto em relação aos demais apresentados nas Tabelas 7 e 8, se dá, pela sua praticidade no desenvolvimento de modelos virtuais, combinando um ambiente de *desktop* para análise iterativa e simultânea, permitindo testes e verificações dos blocos integrados ao modelo durante todas as etapas do processo, o que garante maior integridade ao sistema simulado, dando maior confiabilidade de desenvolvimento do *hardware*.

3.2.5.2 Softwares Complementares

Atividades secundárias resultantes da modelagem principal, abordaram questões como: criação de diagramas elétricos, simulações de RF, mapeamento e análise da topografia do ambiente a ser analisado. Para o atendimento destes fatores, fez-se necessário a utilização de *softwares* complementares que apoiaram em momentos específicos no suporte e validação de resultados.

A lista a seguir apresenta estas ferramentas de apoio, compondo um breve resumo em relação as suas características gerais e aplicações no âmbito desta pesquisa. As ferramentas aqui presentes, estão ordenadas conforme aparecem no texto, sendo todas aplicadas durante a fase de resultados.

- ***Proteus Design Suite Versão 8***, trata-se de um programa proprietário, desenvolvido pela empresa Labcenter Electronics, tem como funcionalidades o projeto e simulação de circuitos eletroeletrônicos, criação de componentes ou blocos para simulação, projetos de placas de circuito impresso aplicados ao desenvolvimento de circuitos integrados. No contexto deste estudo, esta ferramenta foi adotada na

concepção dos diagramas elétricos das estações, que compõem o Subsistema de Telemetria e Comando (T&C).

- ***SolidWorks Versão 2018***, trata-se de um programa proprietário, desenvolvido pela empresa SolidWorks Corporation, *software* de CAD 3d, baseado na elaboração de formas tridimensionais a partir de operações geométricas elementares. Sua adoção neste estudo ocorreu durante o desenvolvimento do esquema detalhado do protótipo da Estação Terrestre (ET), correspondente a Unidade de Condicionamento Funcional (UCF).
- ***Radio Mobile Versão 11.6.6***, ferramenta computacional *open source* desenvolvido por Roger Coudé, seu objetivo é estimar a potência de recepção entre a estação base e seus respectivos clientes, simulando a propagação de RF. Sua aplicação permitiu a análise do Subsistema de Telemetria e Comando, baseado nas informações modeladas para os domínios específicos, permitindo a validação na etapa HIL (*Hardware-In-The-Loop*), do *hardware* proposto, em comparação aos resultados da simulação do modelo virtual.
- ***Google Earth Versão Pro***, ferramenta gratuita desenvolvida pela fabricante Google Inc., permite a análise tridimensional do mapeamento do globo terrestre, a partir de um mosaico de imagens obtidas de diversas fontes de dispositivos aeroespaciais. Este *software*, apoiou na geração de arquivos relacionados a topografia do terreno e espaço desejado, exportando dados específicos que trabalham em conjunto com o *software Radio Mobile*, e fornece aspectos importantes para a análise do espectro de frequência entre a estação base e os respectivos clientes.
- ***Arduino IDE Versão 1.8.9***, ferramenta de desenvolvimento de *software*, *open source* liberado sob a Licença Pública Geral GNU, versão 2. Trata-se de um ambiente de desenvolvimento integrado (IDE, do inglês *Integrated Development Environment*) do Arduino escrito na linguagem de programação Java, usado para escrever e carregar programas com suporte as Linguagens C e C++, usando regras especiais de estruturação de código para placas compatíveis com Arduino. A sua utilização neste estudo se deu durante a etapa de adaptação do código gerado pelo modelo virtual aos aspectos exigidos pela arquitetura lógica, em atendimento as funções mapeadas.
- ***RF Setting Versão 0.70***, *software open source*, desenvolvido pela fabricante EByte, para configuração dos seus dispositivos de radiofrequência. Adotado para a parametrização dos radiotransmissores aplicados neste estudo.
- ***PuTTY Versão 3.45***, *software open source* de emulação de terminal. Aplicado na interface homem-máquina como monitor de comunicação de telemetria e comando durante a operação da ET e da EE.

3.3 Resumo do capítulo

O capítulo 3, explanou modelos de referência aplicados ao desenvolvimento de sistemas, propostos por diferentes autores. A análise de cada uma das abordagens abriu caminho para a compreensão do conhecimento dentro do domínio do projeto do produto, e as variáveis envolvidas referentes à composição de equipes multidisciplinares, sua colaboração no decorrer do processo de projeto simultâneo e à interação nos aspectos do domínio específico que envolvem o sistema.

Na busca por um modelo de referência que atenda à proposta deste estudo, o capítulo destacou a integração dos Modelos MBD e V, o primeiro contribuindo com os testes e verificações, através de simulações em *loop* e o segundo na separação das etapas de desenvolvimento e integração dos domínios específicos. Esta composição somada as ferramentas de *software* colaborativas e de simulação, agrega condições de modelagem para uma aplicação produto a partir de modelos teóricos, garantindo a estimação de parâmetros importantes do sistema, levantamento de potenciais falhas durante o processo de simulação, com base no modelo virtual modelado, o qual reproduz condições similares as de um protótipo real, auxiliando em aspectos para adoção de um processo de desenvolvimento mais rápido e com custos reduzidos.

Nos próximos capítulos, a metodologia híbrida de integração do Modelo V com o MBD, é apresentada dentro do escopo de desenvolvimento de produto de sistemas aplicada ao objetivo do estudo referente a modelagem de um subsistema de comunicação ponto-a-ponto do tipo *half-duplex* dedicado a missões direcionadas para plataformas estratosféricas, suas particularidades são mapeadas com o auxílio dos métodos presentes no Modelo V e as técnicas de testes e verificações, MIL, SIL e HIL, que integram o MBD, afim de compor os resultados e prover as necessidades reais exigidas.

4 Modelagem do Subsistema de Comunicação

Este Capítulo apresenta as etapas do desenvolvimento para o fluxo adaptado do Modelo V, ilustrado na Figura 21, com ênfase no mapeamento e análise dos requisitos funcionais e não-funcionais, definição dos domínios específicos e seus respectivos componentes, e de posse destes elementos a elaboração das arquiteturas correspondentes aos domínios físicos e lógicos. Estas análises permitiram o entendimento das funções que envolvem o subsistema para a composição do modelo virtual e a realização dos testes e verificações que validam a modelagem MBD que será discutida no Capítulo 5.

4.1 Requisitos do subsistema de comunicação

Neste contexto, busca-se levantar as necessidades dos clientes, as quais são agrupadas e eliminadas as possíveis repetições, além daquelas necessidades pouco relevantes para o projeto.

Rozenfeld et al. (2006) orienta após o agrupamento, análise e classificação, que essas necessidades, inicialmente descritas segundo a linguagem dos clientes, podem ser reescritas na forma de requisitos dos clientes. Estes requisitos estão relacionados a aspectos, tais como: desempenho funcional, fatores humanos, propriedades, espaço, confiabilidade, ciclo de vida, recursos e manufatura.

Segundo Curran et al. (2015), os requisitos de um sistema são geralmente classificados em funcionais e não funcionais. Os requisitos funcionais são aqueles que devem ser executados para cumprir os objetivos de operar ou usar o produto. Os requisitos do cliente são expectativas sobre o produto (ou sistema) em termos de missão, objetivos, funções, ambiente e restrições que são apontados pelos próprios clientes das partes interessadas. Esses requisitos são especificados para definir os valores de destino para medir os atributos do produto.

A Tabela 9 apresenta as necessidades levantadas extraídas durante reuniões e conversas com o coordenador e alunos do Laboratório de Simulação e Controle de Sistemas Aeroespaciais (LODESTAR) da Universidade de Brasília (UnB), com o objetivo de entender as expectativas dos envolvidos em relação a um subsistema de comunicação que envolva funções de Telemetria e Comando, onde em resumo, trabalham no seguinte aspecto: A função de Telemetria permite o *downlink*, dos dados de carga útil coletados por sensores (temperatura, umidade, pressão, entre outros) e manutenção da Estação Embarcada (EE)

para a Estação Terrestre (ET), e a função de Comando faz parte do *uplink*, partindo da Estação Terrestre (ET) para a Estação Embarcada (EE), visando o acionamento ou desativação de processos na carga útil, reconfiguração de seus subsistemas para responder às condições da missão, pode incluir subsistemas de comutação e alterar condições de operações de componentes.

Tabela 9 – Necessidades dos usuários.

NECESSIDADES DOS USUÁRIOS
Transmitir dados entre estações
Receber dados entre estações
Receber confirmação de entrega entre estações
Coletar dados provenientes de sensores embarcados
Leve
Compacto
Atender as normas de dimensionamento
Atender as normas de operação
Conectividade com <i>hardwares</i> externos
Compatibilidade com demais componentes do sistema embarcado
Tratamento de falhas na comunicação entre estações
Possuir interface de comandos
Possuir encapsulamento para pacotes de dados
Armazenamento de informações durante a operação
Permitir futuras atualizações dos componentes
Enviar comandos em tempo real
Monitorar a saúde da estação embarcada em tempo real
Transmissão ponto-a-ponto em diferentes cenários
Independente de fontes de energia
Utilizar tecnologia disponível no mercado (COTS)
Possuir tecnologia de baixo custo
Funcionar em ambiente com variações climáticas

Fonte Própria.

Conforme exposto o subsistema de comunicação, conta com atividades de transmissão e recepção de dados, executados pelas estações ET (Estação Terrestre) e EE (Estação Embarcada), o que pode gerar uma certa dificuldade de entendimento no momento do levantamento das necessidades dos usuários, pois condições de Telemetria presentes na EE, possui características próprias, que as diferem das condições de Comando presentes na ET, por este motivo faz-se necessária a especificação individual dentro de cada estação, buscando com isto, um melhor mapeamento das necessidades dos usuários, além de uma

sintonia entre o cliente e as funcionalidades que cada estação poderá proporcionar.

Visando atender as características gerais de funcionamento do Subsistema T&C em separado, faz-se necessária a revisão das necessidades levantadas pelos usuários para o refinamento das exigências e satisfação em relação ao pleno funcionamento dos componentes que compõem o conjunto, sendo assim a Tabela 9, foi revisada e feito a redistribuição das necessidades para cada subsistema, transformando-os em expressões mensuráveis que originaram em requisitos do cliente, conforme apresentado na Tabela 10, chegando-se a fatores similares e específicos no contexto de cada estação, o que conseqüentemente refletirá no requisito do sistema.

Tabela 10 – Requisitos do cliente.

REQUISITOS DO CLIENTE	ET	EE
Transmitir dados	X	X
Receber dados	X	X
Confirmar entrega	X	X
Coletar dados		X
Ser leve	X	X
Ser compacto	X	X
Operar em faixa de frequência regulamentada	X	X
Operar em potência de transmissão regulamentada	X	X
Possuir conectividade	X	X
Encapsular dados	X	X
Ser compatível	X	X
Tratar falhas de comunicação	X	
Possuir interface Homem-Máquina (IHM)	X	
Possuir <i>log</i> de informações		X
Permitir <i>upgrade</i> de componentes	X	X
Operar em tempo real	X	X
Possuir especificações técnicas bem definidas	X	X
Garantir enlace de comunicação em tempo integral	X	X
Utilizar fontes de energia alternativas	X	X
Adotar tecnologia disponível no mercado (COTS)	X	X
Ser de baixo custo	X	X
Ser adaptável		X
Ser controlável	X	X
Ser confiável	X	X
Possuir qualidade	X	X

Fonte Própria.

No MBD, a etapa de requisitos é revista constantemente, através das iterações impostas, durante as fases de testes e verificações, o que garante o refinamento dos requisitos ao longo do desenvolvimento do modelo virtual, minimizando os retrabalhos ou ausência de requisitos, seguindo este preceito os requisitos do Cliente apresentados na Tabela 10, são revistos e refinados novamente, afim de proporcionar uma análise em relação aos critérios de comparação entre os requisitos, para que se possa levantar a relevância de um requisito sobre o outro e então ordená-los por nível de importância. Para esta estratégia de refinamento aplicou-se o diagrama de Mudge.

Curran et al. (2015) destaca que o diagrama de Mudge é uma ferramenta que permite a comparação de função de duas em duas, com o objetivo de ordená-las por relevância. Esta comparação é feita geralmente enumerando as funções como 1,2,3,... n, onde n é o número de funções, posteriormente atribui-se valores para as comparações (SCHUSTER et al., 2015). A Figura 23, apresenta o Gráfico de Pareto gerado pelas comparações realizadas no Diagrama de Mudge. Os requisitos em ordem de relevância no estudo em questão são apresentados detalhadamente no Apêndice A.

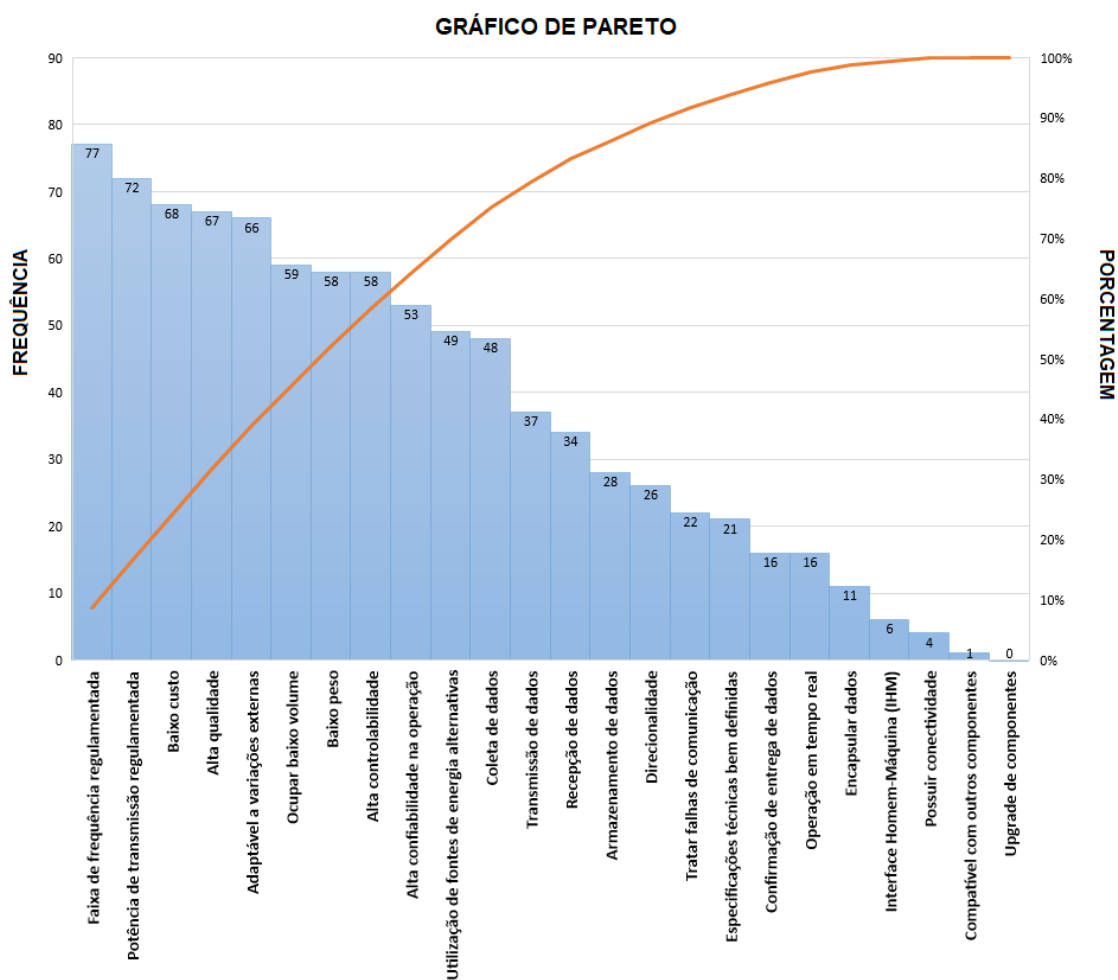


Figura 23 – Gráfico de Pareto.
Fonte Própria.

Com os requisitos do cliente mapeados e hierarquizados, a preocupação foi rastrear o cenário multidisciplinar que envolve o desenvolvimento de sistemas, recaindo nas especificações das diferentes áreas, devido a sua natureza de domínio cruzado presente no cenário do subsistema T&C, a qual envolve elementos da mecânica, eletrônica e *software*, os quais possuem características similares, mas agregam outras específicas que merecem tratamento individual durante as etapas de verificações e testes, recaindo no processo de evolução do eixo y da Matriz de Adoção apresentada na Figura 18.

Nesta etapa os requisitos do cliente foram vinculados as respectivas áreas, buscando iniciar o processo de especificação dos requisitos do sistema e ampliação da visão em relação aos atributos envolvidos e suas finalidades, facilitando assim a quantificação e mensuração dos parâmetros que o sistema deverá ter. Estas especificações, além de atuarem como guias para a geração de soluções para o problema do projeto, fornecem a base sobre a qual serão montados os critérios de avaliação e de tomada de decisão (ROZENFELD et al., 2006), utilizados em breve, durante a construção do modelo virtual.

A Tabela 11 apresenta os elementos das diferentes áreas envolvidas, inseridos em cada estação, realizando um integração entre os requisitos do cliente com os respectivos domínios cruzados, presentes no escopo deste estudo.

Tabela 11 – Vínculo dos requisitos ao domínio cruzado.

REQUISITOS DO CLIENTE	ET			EE		
	1	2	3	1	2	3
Faixa de frequência regulamentada	X	X	X	X	X	X
Potência de transmissão regulamentada	X	X	X	X	X	X
Baixo custo	X	X	X	X	X	X
Alta qualidade	X	X	X	X	X	X
Adaptável a variações externas				X	X	
Ocupar baixo volume		X		X	X	
Baixo peso		X		X	X	
Alta controlabilidade	X	X	X		X	X
Alta confiabilidade na operação	X	X	X	X	X	X
Utilização de fontes de energia alternativas		X			X	
Coleta de dados					X	X
Transmissão de dados	X	X	X	X	X	X
Recepção de dados	X	X	X	X	X	X
Armazenamento de dados		X	X		X	X
Direcionalidade	X					
Tratar falhas de comunicação			X			X
Especificações técnicas bem definidas	X	X	X	X	X	X
Confirmação de entrega de dados	X		X		X	X
Operação em tempo real	X	X	X	X	X	
Encapsular dados			X			X
Interface Homem-Máquina (IHM)		X	X			
Possuir conectividade	X	X		X	X	
Compatível com outros componentes	X	X	X	X	X	X
<i>Upgrade</i> de componentes	X	X	X	X	X	X

LEGENDA	
1	Mecânica
2	Eletrônica
3	Software

Fonte Própria.

No processo de relacionamento entre os requisitos do cliente com os respectivos domínios cruzados, deu-se início a etapa de análise dos requisitos do produto, dito isto, a fase seguinte foi classificar e hierarquizar os requisitos do sistema, quanto à sua quantificação e a intensidade de contribuição, buscando valorar o requisito do sistema e seu impacto no atendimento ao requisito do cliente, no intuito de esclarecer estas condições, foi necessário a unificação da linguagem afim de satisfazer as condições de operação, as quais são apresentadas na Tabela 12 em ordem de prioridade.

Tabela 12 – Classificação e quantificação dos requisitos do sistema.

REQUISITO DO CLIENTE	REQUISITO DO SISTEMA	REQUISITO DO SISTEMA (QUANTIFICÁVEL)
Faixa de frequência regulamentada	Faixa de frequência de operação	Intervalo de frequência de operação
Potência de transmissão regulamentada	Faixa de potência de operação	Intervalo de potência de operação
Baixo custo	Preço	Custo de produção
Alta qualidade	Desempenho operacional	Capacidade máxima de missões
Adaptável a variações externas	Temperatura de operação	Intervalo de temperatura de operação
Ocupar baixo volume	Dimensões	Dimensões do produto
Baixo peso	Peso	Peso do produto
Alta controlabilidade	Controlabilidade	Nível de automação
Alta confiabilidade na operação	Precisão	Taxa de acertos de transmissão/recepção
Utilização de fontes de energia alternativas	Flexibilização	Quantidade de diferentes fontes de energias aceitas
Coleta de dados	Capacidade de informação coletada	Quantidade máxima de sensores conectados
Transmissão de dados	Capacidade de transmissão de dados	Taxa de transmissão de dados
Recepção de dados	Capacidade de recepção de dados	Taxa de recepção de dados
Armazenamento de dados	Capacidade de armazenamento de dados	Quantidade máxima de armazenamento de dados
Direcionalidade	Capacidade de alcance do feixe direcional	Ganho da antena
Tratar falhas de comunicação	Exatidão	Quantidade de pacotes entregues/perdidos
Especificações técnicas bem definidas	Normatização	Quantidade de especificações definidas
Confirmação de entrega de dados	Tempo de Resposta	Tempo de confirmação dos pacotes entregues
Operação em tempo real	Garantia de operação	Distância máxima de operação
Encapsular dados	Compatibilidade de comunicação	Quantidade de protocolos compatíveis
Interface Homem-Máquina (IHM)	Monitoração	Quantidade de interfaces de monitoramento
Possuir conectividade	Compatibilidade de conexão	Quantidade de conectores adotados
Compatível com outros componentes	Versatilidade	Quantidade de fornecedores disponíveis no mercado
Upgrade de componentes	Atualização tecnológica	Tempo de duração do produto

Fonte Própria.

4.1.1 Especificação dos requisitos funcionais

Os requisitos do sistema apresentados na Tabela 12, possuem particularidades que compõem a sua estrutura, um fator preponderante nesta etapa foi a consolidação dos aspectos que motivaram a presença destes neste estudo, desta forma uma especificação mais detalhada se faz essencial para a compreensão e esclarecimento das funções e componentes envolvidos, no intuito de evidenciar as relações com o domínio cruzado e o posicionamento da solução frente ao contexto do problema.

O detalhamento de cada requisito encontra-se estruturado da seguinte forma:

- (a) **Identificador:** apresenta o identificador do requisito funcional dentro do sistema em ordem de prioridade;

- (b) **Nome:** nomenclatura do requisito funcional;
- (c) **Descrição:** apresenta a função principal do requisito;
- (d) **Justificativa:** explica o porque da necessidade dentro do contexto;
- (e) **Vínculo:** relaciona o requisito ao tipo de estação correspondente, nesta etapa existem condições duplicadas, pois um mesmo requisito pode estar presente tanto na ET quanto na EE, devido a sua particularidade de operação;
- (f) **Prioridade:** classificação da necessidade dos requisitos em relação ao seu vínculo, podendo ser do tipo essencial, importante, desejável ou ausente. Os tipos podem ser entendidos como:
 - Essencial é o requisito sem o qual o sistema não entra em funcionamento. Requisitos essenciais são imprescindíveis, ou seja, tem que ser implementados obrigatoriamente.
 - Importante é o requisito sem o qual o sistema entra em funcionamento, mas de forma não satisfatória. Requisitos importantes devem ser implementados, mas, se não forem, o sistema poderá entrar em operação mesmo assim.
 - Desejável é o requisito que não compromete as funcionalidades básicas do sistema, neste caso, o sistema pode operar de forma satisfatória sem ele. Requisitos desejáveis podem ser deixados para versões posteriores do sistema, caso não haja tempo hábil para implementá-los na versão atual.
 - Ausente tipo de prioridade que não pertence ao requisito em questão. Requisitos ausentes não fazem parte do escopo ao qual está sendo vinculado, sua presença foge as funcionalidades especificadas.
- (g) **Domínio cruzado:** Área do processo de desenvolvimento a qual o requisito está envolvido, no âmbito deste estudo estão relacionadas a:
 - Mecânica;
 - Eletrônica;
 - *Software*;
- (h) **Hardware:** Tipo de *hardware* embarcado que representa o requisito, relacionada ao respectivo domínio cruzado do tipo mecânica ou eletrônica;
- (i) **Código de programação:** Funções lógicas que compõem o requisito, relacionada ao respectivo domínio cruzado do tipo *software*;
- (j) **Quantificação:** Valoração do requisito buscando a mensuração do seu desempenho;

A Tabela 13, lista todos os requisitos do sistema levantados até aqui, devidamente referenciados para as suas respectivas modelagens de acordo com as necessidades, as quais são documentadas logo após, onde as especificações de cada requisito presente na lista são detalhadas e devidamente identificadas, sendo composta por um indicador único, formado pela sigla RQF (Requisito Funcional), o número de prioridade dentro do sistema, em ordem hierárquica, o nome do requisito e as condições técnicas envolvidas, conforme as exigências apresentadas anteriormente.

Tabela 13 – Lista de requisitos funcionais do sistema.

REFERÊNCIAS DOS REQUISITOS FUNCIONAIS DO SISTEMA		
ID	REQUISITO DO SISTEMA	REFERÊNCIA
RQF01	FAIXA DE FREQUÊNCIA DE OPERAÇÃO	Tabela 14
RQF02	FAIXA DE POTÊNCIA DE OPERAÇÃO	Tabela 15
RQF03	PREÇO	Tabela 16
RQF04	DESEMPENHO OPERACIONAL	Tabela 17
RQF05	TEMPERATURA DE OPERAÇÃO	Tabela 18
RQF06	DIMENSÕES	Tabela 19
RQF07	PESO	Tabela 20
RQF08	CONTROLABILIDADE	Tabela 21
RQF09	PRECISÃO	Tabela 22
RQF10	FLEXIBILIZAÇÃO	Tabela 23
RQF11	CAPACIDADE DE INFORMAÇÃO COLETADA	Tabela 24
RQF12	CAPACIDADE DE TRANSMISSÃO DE DADOS	Tabela 25
RQF13	CAPACIDADE DE RECEPÇÃO DE DADOS	Tabela 26
RQF14	CAPACIDADE DE ARMAZENAMENTO DE DADOS	Tabela 27
RQF15	CAPACIDADE DE ALCANCE DO FEIXE DIRECIONAL	Tabela 28
RQF16	EXATIDÃO	Tabela 29
RQF17	NORMATIZAÇÃO	Tabela 30
RQF18	TEMPO DE RESPOSTA	Tabela 31
RQF19	GARANTIA DE OPERAÇÃO	Tabela 32
RQF20	COMPATIBILIDADE DE COMUNICAÇÃO	Tabela 33
RQF21	MONITORAÇÃO	Tabela 34
RQF22	COMPATIBILIDADE DE CONEXÃO	Tabela 35
RQF23	VERSATILIDADE	Tabela 36
RQF24	ATUALIZAÇÃO TECNOLÓGICA	Tabela 37

Fonte Própria.

Os requisitos listados na Tabela 13 são documentados individualmente em seguida, analisando cada detalhamento citado anteriormente, em atendimento aos itens da estrutura

proposta e estão distribuídos conforme exposto nas referências.

Tabela 14 – Especificação do requisito - [RQF01].

[RQF01] FAIXA DE FREQUÊNCIA DE OPERAÇÃO					
Descrição	Frequência de comunicação para transmissão e recepção de sinais eletromagnéticos.				
Justificativa	Operação dentro de faixa de frequência regulamentada destinada ao serviço de comunicações via satélite.				
Vínculo		Essencial	Importante	Desejável	Ausente
	ET	X			
	EE	X			
Domínio Cruzado		Mecânica	Eletrônica	Software	
	ET	X	X	X	
	EE	X	X	X	
Hardware	Rádio Transmissor e Antena				
Código de Programação					
Quantificação	Intervalo de frequência de operação em Hertz (Hz)				

Fonte Própria.

Tabela 15 – Especificação do requisito - [RQF02].

[RQF02] FAIXA DE POTÊNCIA DE OPERAÇÃO					
Descrição	Potência de transmissão aplicada na emissão dos sinais eletromagnéticos.				
Justificativa	Operação dentro dos limites de potência regulamentados destinados ao serviço de comunicações via satélite.				
Vínculo		Essencial	Importante	Desejável	Ausente
	ET	X			
	EE	X			
Domínio Cruzado		Mecânica	Eletrônica	Software	
	ET	X	X	X	
	EE	X	X	X	
Hardware	Rádios Transmissores				
Código de Programação					
Quantificação	Intervalo de potência de transmissão em Watts (W) ou Decibel miliwatt (dBm)				

Fonte Própria.

Tabela 16 – Especificação do requisito - [RQF03].

[RQF03] PREÇO					
Descrição	Tecnologia de baixo custo financeiro.				
Justificativa	Adotar metodologia MBD, através de simulação de modelos e aplicar ao produto final tecnologia disponível no mercado (COTS) com custo-benefício satisfatório.				
Vínculo	ET	Essencial	Importante	Desejável	Ausente
	EE	X			
Domínio Cruzado		Mecânica	Eletrônica	Software	
	ET	X	X		X
	EE	X	X		X
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador, Componentes Eletrônicos, Conectores e Softwares.				
Código de Programação					
Quantificação	Custo de produção em reais (R\$)				

Fonte Própria.

Tabela 17 – Especificação do requisito - [RQF04].

[RQF04] DESEMPENHO OPERACIONAL					
Descrição	Estabilidade no funcionamento do sistema.				
Justificativa	Desempenho apropriado dentro das especificações definidas para o sistema, em atendimento as necessidades do cliente.				
Vínculo	ET	Essencial	Importante	Desejável	Ausente
	EE	X			
Domínio Cruzado		Mecânica	Eletrônica	Software	
	ET	X	X		X
	EE	X	X		X
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador, Componentes Eletrônicos, Conectores e Softwares.				
Código de Programação					
Quantificação	Capacidade máxima de missões em quantidade (un).				

Fonte Própria.

Tabela 18 – Especificação do requisito - [RQF05].

[RQF05] TEMPERATURA DE OPERAÇÃO				
Descrição	Adaptação do sistema a intempéries externas, ocasionadas por mudanças climáticas.			
Justificativa	O sistema deve manter o desempenho operacional sob alterações ocasionadas por alterações externas vinculadas ao clima.			
Vínculo	ET			X
	EE	X		
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET	X	X	
	EE	X	X	
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador e Componentes Eletrônicos.			
Código de Programação				
Quantificação	Range de temperatura de operação em graus Celsius (°C).			

Fonte Própria.

Tabela 19 – Especificação do requisito - [RQF06].

[RQF06] DIMENSÕES				
Descrição	Dimensões reduzidas em atendimento as exigências de espaço disponível.			
Justificativa	Os equipamentos do sistema devem possuir dimensões adequadas em atendimento as normas que regem as especificações para satélites de pequeno porte.			
Vínculo	ET			X
	EE	X		
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET		X	
	EE	X	X	
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador e Componentes Eletrônicos.			
Código de Programação				
Quantificação	Dimensões do artefato em milímetros (mm).			

Fonte Própria.

Tabela 20 – Especificação do requisito - [RQF07].

[RQF07] PESO				
Descrição	Massa reduzida em atendimento as exigências de peso disponível.			
Justificativa	O sistema deve possuir massa adequada em sua EE em atendimento as normas que regem as especificações para satélites de pequeno porte.			
Vínculo	ET			X
	EE	X		
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET		X	
	EE	X	X	
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador e Componentes Eletrônicos.			
Código de Programação				
Quantificação	Peso do artefato em kilograma (kg).			

Fonte Própria.

Tabela 21 – Especificação do requisito - [RQF08].

[RQF08] CONTROLABILIDADE				
Descrição	Controle da transmissão, saúde da EE e funcionalidades gerais do sistema.			
Justificativa	O sistema deve permitir condições de controle das possíveis situações de comando integradas.			
Vínculo	ET	X		
	EE		X	
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET	X	X	X
	EE		X	X
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador, Componentes Eletrônicos e Software.			
Código de Programação				
Quantificação	Função Principal			
Quantificação	Nível de automação (%).			

Fonte Própria.

Tabela 22 – Especificação do requisito - [RQF09].

[RQF09] PRECISÃO					
Descrição	Precisão na transmissão, comandos e monitoramento.				
Justificativa	O sistema deve proporcionar alta confiabilidade na operação, que envolve transmissão, comando, recepção, telemetria e monitoramento.				
Vínculo	ET	Essencial	Importante	Desejável	Ausente
	EE	X			
Domínio Cruzado		Mecânica	Eletrônica	Software	
	ET	X	X	X	
	EE	X	X	X	
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador, Componentes Eletrônicos.				
Código de Programação	Função Principal				
Quantificação	Taxa de acertos de transmissão/recepção (%).				

Fonte Própria.

Tabela 23 – Especificação do requisito - [RQF10].

[RQF10] FLEXIBILIZAÇÃO					
Descrição	Fontes de energia alternativas.				
Justificativa	Permitir alternativas no fornecimento de energia para alimentação do sistema, oriundas de baterias, placas solares, entre outros.				
Vínculo	ET	Essencial	Importante	Desejável	Ausente
	EE	X			
Domínio Cruzado		Mecânica	Eletrônica	Software	
	ET		X		
	EE		X		
Hardware	Bateria e Placa Solar				
Código de Programação					
Quantificação	Fontes de energias aceitas em quantidade (un).				

Fonte Própria.

Tabela 24 – Especificação do requisito - [RQF11].

[RQF11] CAPACIDADE DE INFORMAÇÃO COLETADA				
Descrição	Capacidade de coleta de dados proveniente de sensores.			
Justificativa	Permitir a conexão dos diferentes sensores embarcados, para monitoramento da saúde e demais informações do sistema.			
Vínculo	ET			X
	EE	X		
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET			
	EE		X	X
Hardware	Microcontrolador, Bateria, Componentes Eletrônicos.			
Código de Programação	Função Coleta de Dados			
Quantificação	Sensores conectados aceitos em quantidade (un).			

Fonte Própria.

Tabela 25 – Especificação do requisito - [RQF12].

[RQF12] CAPACIDADE DE TRANSMISSÃO DE DADOS				
Descrição	Capacidade de transmissão de dados.			
Justificativa	Análise da taxa de transmissão de dados entre as estações.			
Vínculo	ET	X		
	EE	X		
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET	X	X	X
	EE	X	X	X
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador, Componentes Eletrônicos.			
Código de Programação	Função Transmissão			
Quantificação	Taxa de transmissão de dados em bits por segundo (bps).			

Fonte Própria.

Tabela 26 – Especificação do requisito - [RQF13].

[RQF13] CAPACIDADE DE RECEPÇÃO DE DADOS				
Descrição	Capacidade de recepção de dados.			
Justificativa	Análise da taxa de recepção de dados.			
Vínculo	ET	Essencial	Importante	Desejável Ausente
	EE	X		
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET	X	X	X
	EE	X	X	X
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador, Componentes Eletrônicos.			
Código de Programação	Função Recepção			
Quantificação	Taxa de recepção de dados em bits por segundo (bps).			

Fonte Própria.

Tabela 27 – Especificação do requisito - [RQF14].

[RQF14] CAPACIDADE DE ARMAZENAMENTO DE DADOS				
Descrição	Capacidade de armazenamento de dados.			
Justificativa	Possuir <i>log</i> de informações para armazenamento das informações coletadas durante a operação do sistema.			
Vínculo	ET	Essencial	Importante	Desejável Ausente
	EE	X	X	
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET		X	X
	EE		X	X
Hardware	Bateria, Microcontrolador, Shield Data logging.			
Código de Programação	Função Log			
Quantificação	Capacidade de armazenamento de dados em bytes (GB).			

Fonte Própria.

Tabela 28 – Especificação do requisito - [RQF15].

[RQF15] CAPACIDADE DE ALCANCE DO FEIXE DIRECIONAL				
Descrição	Capacidade de alcance do feixe direcional			
Justificativa	Possuir feixe direcional, para melhor desempenho da capacidade do canal de comunicação durante a operação do sistema.			
Vínculo	ET	X		
	EE			X
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET	X		
	EE	X		
Hardware		Antena		
Código de Programação				
Quantificação	Ganho da antena em décibéis (dBi).			

Fonte Própria.

Tabela 29 – Especificação do requisito - [RQF16].

[RQF16] EXATIDÃO				
Descrição	Tratamento de falhas de comunicação durante o processo de transmissão do sistema.			
Justificativa	Monitorar a quantidade de pacotes entregues com sucesso e pacotes perdidos, durante o processo de transmissão.			
Vínculo	ET	X		
	EE	X		
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET			X
	EE			X
Hardware				
Código de Programação	Função TimeOut			
Quantificação	Pacotes entregues ou perdidos em quantidade (un).			

Fonte Própria.

Tabela 30 – Especificação do requisito - [RQF17].

[RQF17] NORMATIZAÇÃO				
Descrição	Definição das especificações técnicas.			
Justificativa	Normalizar para devida documentação das especificações técnicas que compõem o sistema.			
Vínculo	ET	Essencial	Importante	Desejável Ausente
	EE	X		
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET	X	X	X
	EE	X	X	X
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador, Shield Data logging, Componentes Eletrônicos.			
Código de Programação				
Quantificação	Especificações técnicas definidas em quantidade (un).			

Fonte Própria.

Tabela 31 – Especificação do requisito - [RQF18].

[RQF18] TEMPO DE RESPOSTA				
Descrição	Tempo de resposta para confirmação de entrega de pacotes.			
Justificativa	Mensurar o tempo de envio dos pacotes, para uma análise do atraso durante a operação de transmissão do sistema.			
Vínculo	ET	Essencial	Importante	Desejável Ausente
	EE		X	
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET	X	X	X
	EE	X	X	X
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador, Componentes Eletrônicos.			
Código de Programação				
Quantificação	Tempo de confirmação dos pacotes entregues em segundos (s).			

Fonte Própria.

Tabela 32 – Especificação do requisito - [RQF19].

[RQF19] GARANTIA DE OPERAÇÃO				
Descrição	Garantia da operação em tempo real.			
Justificativa	Analisar a distância máxima de comunicação suportada pelo sistema.			
Vínculo	ET	Essencial	Importante	Desejável Ausente
	EE	X		
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET	X	X	
	EE	X	X	
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador, Componentes Eletrônicos.			
Código de Programação				
Quantificação	Distância máxima de comunicação em quilômetros (km).			

Fonte Própria.

Tabela 33 – Especificação do requisito - [RQF20].

[RQF20] COMPATIBILIDADE DE COMUNICAÇÃO				
Descrição	Encapsulamento de dados em pacotes durante as etapas de transmissão e recepção.			
Justificativa	Possuir protocolo aplicado a sistemas de comunicação via satélite.			
Vínculo	ET	Essencial	Importante	Desejável Ausente
	EE	X		
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET			X
	EE			X
Hardware				
Código de Programação	Protocolos: Serial, I2C, Modbus, X.25, entre outros			
Quantificação	Protocolos compatíveis em quantidade (un).			

Fonte Própria.

Tabela 34 – Especificação do requisito - [RQF21].

[RQF21] MONITORAÇÃO					
Descrição	Interface Homem-Máquina para monitoramento do sistema.				
Justificativa	Possuir visores e entradas para conexões externas via notebook, afim de proporcionar o monitoramento durante o período de operação do sistema.				
Vínculo	ET	Essencial	Importante	Desejável	Ausente
	EE				X
Domínio Cruzado	ET	Mecânica	Eletrônica	Software	
	EE		X		X
Hardware	Microcontrolador, Display LCD				
Código de Programação	Função Display LCD				
Quantificação	Interfaces de monitoramento em quantidade (un).				

Fonte Própria.

Tabela 35 – Especificação do requisito - [RQF22].

[RQF22] COMPATIBILIDADE DE CONEXÃO					
Descrição	Portas de conexão e tipos de conectores adotados.				
Justificativa	Atender as diferentes portas de conexão e possuir acoplamentos para os diferentes tipos de conectores adotados pelos equipamentos do sistema.				
Vínculo	ET	Essencial	Importante	Desejável	Ausente
	EE		X		
Domínio Cruzado	ET	Mecânica	Eletrônica	Software	
	EE	X	X		
Hardware	USB fêmea e macho, Serial, Cabo PigTail (Rgc58 + Sma macho), P4 fêmea e macho, Tamiya fêmea e macho e Dupont fêmea e macho.				
Código de Programação					
Quantificação	Conectores adotados em quantidade (un).				

Fonte Própria.

Tabela 36 – Especificação do requisito - [RQF23].

[RQF23] VERSATILIDADE				
Descrição	Compatibilidade com componentes de outras marcas.			
Justificativa	Ser compatível com outros componentes e <i>hardwares</i> de outros fornecedores.			
Vínculo	ET		X	
	EE		X	
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET	X	X	X
	EE	X	X	X
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador, Componentes Eletrônicos, Conectores.			
Código de Programação	Função Principal			
Quantificação	Produtos concorrentes disponíveis no mercado em quantidade (un).			

Fonte Própria.

Tabela 37 – Especificação do requisito - [RQF24].

[RQF24] ATUALIZAÇÃO TECNOLÓGICA				
Descrição	Upgrade e atualizações tecnológicas futuras.			
Justificativa	Ser capaz de sofrer atualizações no sistema em geral, conforme o avanço da tecnologia.			
Vínculo	ET			X
	EE			X
Domínio Cruzado		Mecânica	Eletrônica	Software
	ET	X	X	X
	EE	X	X	X
Hardware	Rádio Transmissor, Antena, Bateria, Microcontrolador, Componentes Eletrônicos, Conectores.			
Código de Programação	Função Principal e Funções Auxiliares			
Quantificação	Tempo de duração do produto em anos.			

Fonte Própria.

As especificações documentadas, ampliaram a visão interna da caixa preta que

envolve cada requisito funcional, com destaque para as condições de funcionamento que forneceram um entendimento amplo em relação as características que devem atender cada necessidade imposta pelo Cliente, neste caso estas características vinculam as arquiteturas específicas da ET e EE aos domínios cruzados compostos pelos tipos de componentes físicos (*hardware*) ou lógico (*software*), com uma certa similaridade e conseqüente duplicidade no atendimento aos parâmetros como no caso dos requisitos em que o mesmo domínio cruzado esta presente em ambas estações com prioridades idênticas de classificação. Destaque para: RQF01 (Tabela 14), RQF02 (Tabela 15), RQF03 (Tabela 16), RQF04 (Tabela 17), RQF09 (Tabela 22), RQF10 (Tabela 23), RQF12 (Tabela 25), RQF13 (Tabela 26), RQF16 (Tabela 29), RQF17 (Tabela 30), RQF19 (Tabela 32), RQF20 (Tabela 33), RQF22 (Tabela 35), RQF23 (Tabela 36) e RQF24 (Tabela 37), onde buscou-se neste estudo atender a cada um destes requisitos citados, por serem considerados amplos pela sua presença mútua em ambos os vínculos.

Os demais requisitos em que as prioridades dentro de cada vínculo possuem valorações diferentes, como por exemplo em: RQF05 (Tabela 18), RQF06 (Tabela 22), RQF07 (Tabela 20), RQF08 (Tabela 21), RQF11 (Tabela 24), RQF14 (Tabela 27), RQF15 (Tabela 28), RQF18 (Tabela 31) e RQF21 (Tabela 34), nestes casos a ênfase de atendimento no modelo proposto ficou para os requisitos com maiores prioridades, deixando os demais requisitos para futuras atualizações do escopo.

Estes requisitos citados com vínculos diferentes, apesar de em alguns casos eles serem considerados desejáveis ou importantes, suas implementações poderiam trazer custos adicionais e também necessidades mais amplas que podem ser atendidas a princípio apenas para uma estação não perdendo tanto em desempenho do sistema, como exemplo, vale uma análise para o requisito RQF15 (Tabela 28), referente à capacidade de alcance do feixe direcional, o qual está ligado à necessidade de direcionalidade da antena, onde dentro do cenário deste estudo busca-se atender na ET, por meio do seu domínio cruzado do tipo mecânico, com a inclusão de uma antena que possua dimensões maiores, mas com um nível de ganho e espectro diretivo maior, afim de atingir o alvo, que neste caso é a plataforma estratosférica.

Pensando no cenário da EE para o mesmo requisito, abre-se mão da particularidade direcional utilizando-se uma antena do tipo omnidirecional, ou seja, com poder de irradiação do espectro de sinal em 360°, mas com baixo ganho e conseqüente nível de diretividade inexistente, em contrapartida com custo e dimensões físicas menores, ficando esta parte a cargo do nível de controlabilidade e alcance da antena da ET. Vale ressaltar que aplicação de antenas a EE com tamanho reduzido, alto ganho e diretividade, demandam além de maior custo, condições de projetos mais específicas e complexas, direcionadas ao estudo da teoria de antenas, o que acarreta uma análise profunda neste contexto, fugindo do âmbito desta pesquisa.

4.1.2 Especificação dos requisitos não-funcionais

Em decorrência das necessidades dos usuários, que culminou na lista de requisitos funcionais do produto e das iterações de verificação durante as etapas do MBD, deve-se destacar os domínios cruzados compostos pelos tipos de componentes lógicos (*software*), representados pelas funções, relacionando as funcionalidades agregadas, em relação ao comportamento do sistema no âmbito computacional. Assim, em consideração às funcionalidades lógicas, faz-se necessário uma listagem dos requisitos não-funcionais, apresentados na Tabela 38 os quais agregam ao sistema questões de usabilidade, desempenho, segurança, portabilidade e organizacional, variáveis que permeiam a questão do *software* embarcado.

Tabela 38 – Lista de requisitos não-funcionais.

REFERÊNCIAS DOS REQUISITOS NÃO-FUNCIONAIS		
ID	REQUISITO NÃO-FUNCIONAL DO SISTEMA	REFERÊNCIA
RQNF01	FACILIDADE DE USO	Tabela 39
RQNF02	PARAMETRIZAÇÃO DOS DADOS	Tabela 40
RQNF03	TEMPO DE RESPOSTA	Tabela 41
RQNF04	BIBLIOTECAS	Tabela 42
RQNF05	SISTEMA OPERACIONAL	Tabela 43
RQNF06	ENCAPSULAMENTO DE DADOS	Tabela 44
RQNF07	PERMISSÃO DE ACESSO	Tabela 45
RQNF08	PROTOCOLO DE COMUNICAÇÃO	Tabela 46
RQNF09	COMANDO REMOTO	Tabela 47
RQNF10	MEMÓRIA DE ARMAZENAMENTO	Tabela 48
RQNF11	MEMÓRIA RAM	Tabela 49
RQNF12	INTERFACE MICROCONTROLADA	Tabela 50
RQNF13	LINGUAGEM DE PROGRAMAÇÃO	Tabela 51

Fonte Própria.

A Tabela 38, lista os requisitos não-funcionais do sistema, devidamente referenciados por tipo para as suas respectivas modelagens de acordo com as exigências impostas pelas funcionalidades computacionais, documentadas e devidamente identificadas por um indicador único, formado pela sigla RQNF (Requisito Não-funcional), o número de prioridade dentro do sistema, seguindo a estrutura apresentada abaixo:

- (a) **Identificador:** apresenta o identificador do requisito não-funcional dentro do sistema;
- (b) **Nome:** nomenclatura do requisito não-funcional;
- (c) **Descrição:** apresenta a função principal do requisito não-funcional;

- (d) **Tipo:** apresenta o posicionamento do requisito não-funcional em relação as seguintes questões:
- Usabilidade;
 - Desempenho;
 - Segurança;
 - Portabilidade;
 - Organizacional;
- (e) **Vínculo:** relaciona o requisito não-funcional ao tipo de estação correspondente, nesta etapa existem condições duplicadas, pois um mesmo requisito pode esta presente tanto na ET quanto na EE, devido a sua particularidade de operação;
- (f) **Prioridade:** classificação da necessidade dos requisitos não-funcionais em relação ao seu vínculo, podendo ser do tipo essencial, importante, desejável ou ausente. Os tipos podem ser entendidos como:
- Essencial é o requisito sem o qual o sistema não entra em funcionamento. Requisitos essenciais são imprescindíveis, ou seja tem que ser implementados obrigatoriamente.
 - Importante é o requisito sem o qual o sistema entra em funcionamento, mas de forma não satisfatória. Requisitos importantes devem ser implementados, mas, se não forem, o sistema poderá entrar em operação mesmo assim.
 - Desejável é o requisito que não compromete as funcionalidades básicas do sistema, neste caso, o sistema pode operar de forma satisfatória sem ele. Requisitos desejáveis podem ser deixados para versões posteriores do sistema, caso não haja tempo hábil para implementá-los na versão atual.
 - Ausente tipo de prioridade que não pertence ao requisito em questão. Requisitos ausentes não fazem parte do escopo ao qual esta sendo vinculado, sua presença foge as funcionalidades especificadas.
- (g) **Código de programação:** Funções lógicas que auxiliam o requisito não-funcional para a sua concretização, estas funções foram listadas nas especificações dos requisitos, sendo que as demais funções com excessão da principal são consideradas como funções auxiliares e estão classificadas da seguinte forma:
- Função Principal;
 - Função Coleta Dados;
 - Função Transmissão;
 - Função Recepção;

- Função Log;
- Função TimeOut;
- Função Confirmação de Entrega;
- Função Display LCD.

Os requisitos listados na Tabela 38 são documentados individualmente em seguida, analisando cada detalhamento citado anteriormente, em atendimento aos itens da estrutura proposta e estão distribuídos conforme exposto nas referências.

Tabela 39 – Especificação do requisito não-funcional - [RQNF01].

[RQNF01] FACILIDADE DE USO					
Descrição		O sistema tem que ser explicativo, comentado, dinâmico, possuir para cada função, campos e ações, além de caixa de texto ou aviso de erros.			
Tipo		Usabilidade	Desempenho	Segurança	Organizacional
		X			
		Essencial	Importante	Desejável	Ausente
Vínculo	ET	X			
	EE	X			
Código de Programação		Função Principal e Funções Auxiliares			

Fonte Própria.

Tabela 40 – Especificação do requisito não-funcional - [RQNF02].

[RQNF02] PARAMETRIZAÇÃO DOS DADOS					
Descrição		O sistema deve possuir campos devidamente identificados, para preenchimento dos dados pré configurados que demandam parâmetros provenientes do modo de operação do sistema, como: frequência de operação, potência de operação, habilitação de portas lógicas, canal de comunicação, endereçamentos de origem e destino, entre outros.			
Tipo		Usabilidade	Desempenho	Segurança	Organizacional
		X			
		Essencial	Importante	Desejável	Ausente
Vínculo	ET	X			
	EE	X			
Código de Programação		Função Principal			

Fonte Própria.

Tabela 41 – Especificação do requisito não-funcional - [RQNF03].

[RQNF03] TEMPO DE RESPOSTA				
Descrição	O sistema deve possuir monitoramento do tempo de transmissão e recepção, informando ao transmissor, caso o tempo de resposta da confirmação de entrega atrase após intervalo estipulado nos parâmetros do sistema pelo usuário.			
Tipo	Usabilidade	Desempenho	Segurança	Organizacional
		X		
		Essencial	Importante	Desejável
Vínculo	ET	X		
	EE	X		
Código de Programação	Função TimeOut			

Fonte Própria.

Tabela 42 – Especificação do requisito não-funcional - [RQNF04].

[RQNF05] BIBLIOTECAS				
Descrição	O sistema deve atender de forma íntegra as bibliotecas exigidas para o correto funcionamento do <i>hardware</i> embarcado, por meio das bibliotecas adequadas, exigidas para atendimento a correta operação das funções auxiliares, como: visualização em LCD, transmissão e recepção de dados, log em cartão de memória, coleta de dados e protocolos de comunicação envolvidos.			
Tipo	Usabilidade	Desempenho	Segurança	Organizacional
				X
		Essencial	Importante	Desejável
Vínculo	ET	X		
	EE	X		
Código de Programação	Função Principal			

Fonte Própria.

Tabela 43 – Especificação do requisito não-funcional - [RQNF05].

[RQNF05] SISTEMA OPERACIONAL				
Descrição	O sistema deve ser acessado em sistema operacional Windows, Linux ou IOS, e ser compatível com versões atualizadas da interface de desenvolvimento (IDE) e interface de comunicação serial (putty, ssh).			
Tipo	Usabilidade	Desempenho	Segurança	Organizacional
				X
	Essencial	Importante	Desejável	Ausente
Vínculo	ET	X		
	EE	X		
Código de Programação	Função Principal e Funções Auxiliares			

Fonte Própria.

Tabela 44 – Especificação do requisito não-funcional - [RQNF06].

[RQNF06] ENCAPSULAMENTO DE DADOS				
Descrição	O sistema deve ser capaz de encapsular os dados manipulados em pacotes adequados para realização do processo de transmissão na comunicação, contemplando: endereço de origem e destino, contador de mensagens enviadas e informações adicionais referentes à mensagem desejada.			
Tipo	Usabilidade	Desempenho	Segurança	Organizacional
	X			
	Essencial	Importante	Desejável	Ausente
Vínculo	ET	X		
	EE	X		
Código de Programação	Função Principal, Função Transmissão, Função Recepção, Função TimeOut, Função Confirmação de Entrega.			

Fonte Própria.

Tabela 45 – Especificação do requisito não-funcional - [RQNF07].

[RQNF07] PERMISSÃO DE ACESSO					
Descrição		Sistema deve atender as normas de segurança de acesso, autorizando o acesso a dados e sistema IHM somente através de <i>login</i> e senhas de acesso.			
Tipo		Usabilidade	Desempenho	Segurança	Organizacional
				X	
		Essencial	Importante	Desejável	Ausente
Vínculo	ET			X	
	EE			X	
Código de Programação		Função Principal			

Fonte Própria.

Tabela 46 – Especificação do requisito não-funcional - [RQNF08].

[RQNF08] PROTOCOLO DE COMUNICAÇÃO					
Descrição		Sistema deve operar com protocolo de comunicação compatível as exigências e integração com sistemas externos.			
Tipo		Usabilidade	Desempenho	Segurança	Organizacional
					X
		Essencial	Importante	Desejável	Ausente
Vínculo	ET	X			
	EE	X			
Código de Programação		Função Principal			

Fonte Própria.

Tabela 47 – Especificação do requisito não-funcional - [RQNF09].

[RQNF09] COMANDO REMOTO					
Descrição		Sistema tem que possuir IHM de comando para controle da saúde do satélite, como: ejeção, acionamento e/ou desligamento de sensores e atuadores.			
Tipo		Usabilidade	Desempenho	Segurança	Organizacional
		X			
		Essencial	Importante	Desejável	Ausente
Vínculo	ET	X			
	EE	X			
Código de Programação		Função Principal e funções auxiliares			

Fonte Própria.

Tabela 48 – Especificação do requisito não-funcional - [RQNF10].

[RQNF10] MEMÓRIA DE ARMAZENAMENTO					
Descrição		Sistema tem que possuir memória suficiente para armazenamento de dados durante a missão de no mínimo 1GB em cartão SD.			
Tipo		Usabilidade	Desempenho	Segurança	Organizacional
		X			
		Essencial	Importante	Desejável	Ausente
Vínculo	ET			X	
	EE	X			
Código de Programação		Função Principal, Função Log, Função Coleta Dados			

Fonte Própria.

Tabela 49 – Especificação do requisito não-funcional - [RQNF11].

[RQNF11] MEMÓRIA RAM					
Descrição		Sistema requer um mínimo de 2KB de memória RAM para ambas as estações.			
Tipo		Usabilidade	Desempenho	Segurança	Organizacional
		X			
		Essencial	Importante	Desejável	Ausente
Vínculo	ET	X			
	EE	X			
Código de Programação		Função Principal			

Fonte Própria.

Tabela 50 – Especificação do requisito não-funcional - [RQNF12].

[RQNF12] INTERFACE MICROCONTROLADA					
Descrição		Sistema requer uma interface com os requisitos mínimos de um Arduino Uno ou placas com desempenho acima desta como Raspberry, atendendo as normas de dimensões aceitas.			
Tipo		Usabilidade	Desempenho	Segurança	Organizacional
					X
		Essencial	Importante	Desejável	Ausente
Vínculo	ET	X			
	EE	X			
Código de Programação		Função Principal e Funções Auxiliares			

Fonte Própria.

Tabela 51 – Especificação do requisito não-funcional - [RQNF13].

[RQNF13] LINGUAGEM DE PROGRAMAÇÃO					
Descrição		Sistema requer uso de linguagem de programação apropriada as exigências das IDE's adotadas pelas interfaces microcontroladas, como: Linguagem C.			
Tipo		Usabilidade	Desempenho	Segurança	Organizacional
					X
		Essencial	Importante	Desejável	Ausente
Vínculo	ET	X			
	EE	X			
Código de Programação		Função Principal e Funções Auxiliares			

Fonte Própria.

O detalhamento dos requisitos não-funcionais, apresentaram o desdobramento dos requisitos funcionais do sistema, em relação a questão comportamental que envolve os componentes computacionais em que novas condições em relação a exigências organizacionais agregaram a modelagem do sistema, surgindo conceitos técnicos ainda não explorados, com destaque para: RQNF05 (Tabela 43), o qual vincula a exigência de bibliotecas adequadas para o pleno funcionamento do *hardware*, que estará envolvido, como por exemplo: biblioteca SD.h aplicada a componentes que envolvem integração de interface para cartão SD, com o objetivo de gravação de dados, ou ainda a biblioteca LiquidCrystalI2C que envolve integração com interfaces para visor LCD com 2 colunas e 16 linhas. O requisito RQNF12 (Tabela 50), referente a escolha da interface microcontrolada, cita como exemplo, as interfaces Arduino e Raspberry, contribuindo para a construção da arquitetura do sistema e o RQNF13 (Tabela 51), que define a linguagem de programação, a qual acompanhará a interface de desenvolvimento adequada para a interface microcontrolada adotada.

Os requisitos RQNF10 (Tabela 48) e RQNF11 (Tabela 49), estabelecem parâmetros mínimos de funcionamento, limitando condições que possam prejudicar o desempenho do sistema. Os demais requisitos se limitam ao atendimento de exigências que envolvem componentes lógicos, contribuindo para o aprimoramento das funções adotadas para o correto funcionamento do sistema, estes sendo implementados via *software*, com destaque para todos os demais requisitos.

4.2 Arquitetura do subsistema de comunicação

A arquitetura do subsistema busca esclarecer os domínios específicos e suas integrações, presentes na base do modelo V tradicional, apresentado na Figura 16. Nesta etapa ocorre um envolvimento multidisciplinar, pois começa a existir maior coesão na interdependência entre as áreas envolvidas e seus profissionais, afim de alinhar às necessidades e

requisitos, a um escopo viável para o que se quer desenvolver.

4.2.1 Arquitetura física

A arquitetura do subsistema de comunicação no que tange a estrutura física, foi especificada nos requisitos funcionais, em concordância com as características que vinculam as condições de operação das estações ET e EE, aos domínios cruzados específicos para o caso deste estudo, correspondentes a mecânica e eletrônica, compostos pelos tipos de componentes físicos (*hardware*), os quais se encontram distribuídos da seguinte forma:

- **Radiotransmissor**, conhecido também como Transceptor ou Transceiver, dispositivo que combina funções de transmissão e recepção em um único componente, contemplando características específicas de frequência, canal de comunicação e potência;
- **Interface microcontrolada**, são dispositivos que contam com periféricos integrados permitindo a execução de várias funções, além da extensão por meio de portas lógicas digitais e analógicas para controle de periféricos externos. São constituídos de um pequeno processador (CPU, do Inglês *Central Processing Unit*), memória de armazenamento de programa, memória de armazenamento de variáveis, periféricos de comunicação e de forma similar a um computador, podem ser programados por meio de linguagem de programação, como por exemplo, Linguagem C;
- **Antenas**, em síntese são um meio para irradiar e receber ondas de rádio, sua arquitetura é influenciada pelas técnicas de diretividade adotadas, onde fatores como frequência de operação e o ganho desejado, apresentada na Equação 2.4 são fundamentais para definir seu formato e dimensões, esta última influenciada diretamente pelo comprimento da onda eletromagnética, que é calculada pela Equação 2.5, onde na prática comprimentos de onda maiores exigem antenas com tamanhos elevados;
- **Baterias**, dispositivos que fornecem corrente elétrica na forma contínua para alimentação de inúmeros dispositivos eletrônicos, onde a sua geração de energia é desenvolvida por meio de uma reação química, sendo esta reação que caracteriza o tipo da bateria, as mais tradicionais são compostas por: Níquel-Cádmio (Ni-CD), Hidreto metálico de Níquel (Ni-MH), Íon-Lítio, entre outras;
- **Componentes eletrônicos**, periféricos inseridos ao sistema que auxiliam em tarefas específicas, podem estar relacionados à questão da regulação no fornecimento de energia, como por exemplo, Reguladores de tensão (LM2596) e principalmente agregados à interface microcontrolada, como por exemplo, placas para armazenamento de dados (Módulos *Log*), comunicação de dados (Módulos Ethernet), rastreamento (Módulos GPS), ou ainda funcionarem como conversores para comunicação entre

portas distintas, com objetivos de parametrizar interfaces, como por exemplo, o conversor USB-Serial Tll, adotado neste estudo, no processo de parametrização, via porta USB, dos componentes eletrônicos responsáveis pelo sistema de radiotransmissão. Os componentes citados são popularmente conhecidos como *Shields* ou módulos eletrônicos, costumam vir habilitados e compatíveis a interface microcontrolada a qual eles foram projetados, ou podem também serem implementados do zero, atendendo as especificações necessárias.

A partir das especificações técnicas listadas, referentes aos componentes físicos (*hardware*), um diagrama de blocos inicial é apresentado na Figura 24, contemplando a posição de cada elemento dentro do subsistema de comunicação, afim de inserir dentro do contexto geral as estações terrestre (ET) e embarcada (EE).

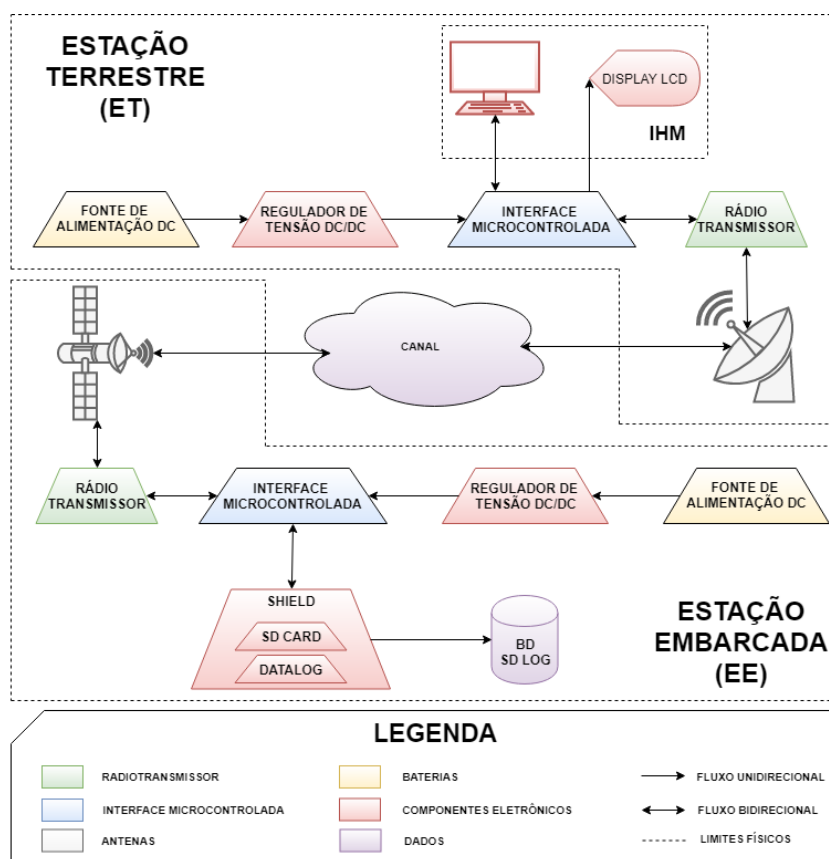


Figura 24 – Diagrama de blocos geral.
Fonte Própria.

O diagrama de blocos apresentado na Figura 24, pode então ser vinculado aos domínios específicos, permitindo um resumo direto dos elementos que envolvem cada domínio, os quais foram detalhados durante a etapa de requisitos do subsistema e estão sintetizados na Figura 25, permitindo um melhor entendimento sobre esta questão.

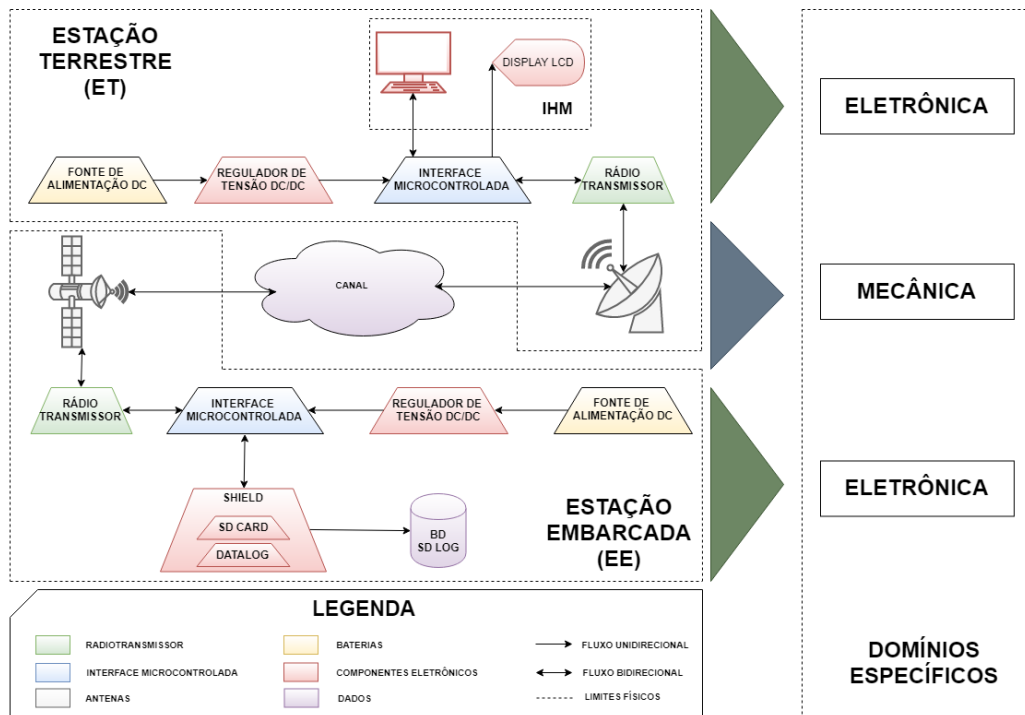


Figura 25 – Domínio específico no diagrama de blocos.
Fonte Própria.

Observa-se na Figura 25, que o domínio Eletrônica está presente em ambas as estações ET e EE, neste caso os respectivos componentes em duplicidade deverão incorporar modelos similares, afim de evitar incompatibilidades na comunicação e conseqüente prejuízo ao desempenho do sistema. Vale ressaltar que algumas diferenças em termos específicos deverão ser consideradas, principalmente nas condições que envolvem parâmetros lógicos e de endereçamento, além de padrões de operação e integração com outros componentes, apropriados para os propósitos de cada estação, com destaque para a existência de um Visor LCD para a ET e um Log de armazenamento de dados na EE. O domínio Mecânica, envolve os componentes do tipo antenas, os quais possuem características distintas no âmbito das estações, as quais serão explanadas em breve.

4.2.2 Mapa tecnológico - Domínio eletrônica

O mapa tecnológico busca vincular as tecnologias envolvidas dentro do sistema, aos seus respectivos fabricantes, no âmbito do cenário estudado. A proposta é prover componentes do tipo COTS de baixo custo, que atendam condições e parâmetros de operação desejados para um subsistema de comunicação via satélite, sendo assim, após o detalhamento dos requisitos realizados anteriormente e a conseqüente construção da arquitetura macro explorada na Figura 24, resultando em um posicionamento em relação aos domínios específicos, envolvendo os componentes físicos, complementados na Figura 25, faz-se necessário o mapeamento tecnológico, para auxílio na escolha das interfaces de

operação que atenderão as especificações exigidas para o subsistema em desenvolvimento.

Pensando no âmbito dos domínios específicos a Figura 26 apresenta os respectivos fabricantes que envolvem o domínio Eletrônica, dentro do respectivo tipo.

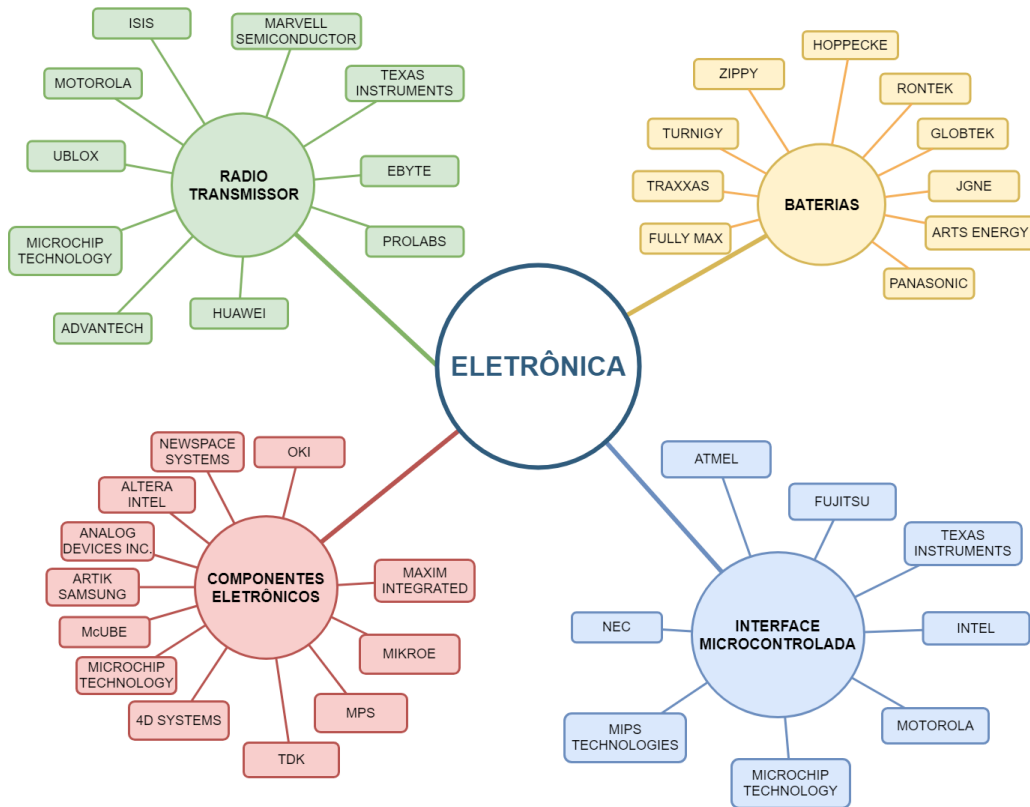


Figura 26 – Mapa de fabricantes - Domínio eletrônico.
Fonte Própria.

O mapa tecnológico do domínio eletrônico, apresentado na Figura 26, listou os diversos fabricantes dos diferentes tipos de *hardwares* envolvidos neste estudo, devido a ampla gama, a pesquisa estará limitada a apenas três modelos de componentes físicos, escolhidos conforme padrões levantados durante a etapa de requisitos como: peso, dimensões, temperatura de operação, compatibilidade de conexão, faixas de operação, além de serem facilmente encontrados no mercado local e com preços acessíveis, visando a implementação do protótipo futuro, após os testes e verificações do modelo virtual.

Buscando atender estes requisitos deu-se andamento a definição do modelo mais apropriado para este estudo, onde iterações de verificação em sintonia com o MBD foram realizadas afim de alcançar um modelo de arquitetura viável, neste caso as iterações foram aplicadas para comparações entre os componentes escolhidos. Deu-se início então, à definição das Interfaces Microcontroladas, a opção em iniciar por este componente se fez necessária, devido ao seu grau de importância dentro do sistema, a qual agrega funcionalidades de controle, processamento, memória, *clock* de operação, portas lógicas digitais e analógicas, portas de comunicação, entre outros fatores, que fazem deste componente

o cérebro do sistema, sendo assim a sua definição inicial ditará a escolha das demais interfaces em termos de compatibilidade, desempenho e operação.

4.2.2.1 Interface Microcontrolada

A maioria dos microcontroladores, estão integrados a uma placa *onboard*, um circuito embarcado ou um kit de desenvolvimento, dentro do contexto deste estudo, como o seu enfoque não é o desenvolvimento de placas eletrônicas, mas sim a aquisição de placas comerciais (COTS) que se enquadrem as exigências do sistema, buscou-se modelos de kit's de desenvolvimento com interfaces microcontroladas integradas ao circuito. A opção por estes kit's se dá pela sua flexibilidade, controlabilidade, programação intuitiva e compatibilidade com diversos módulos eletrônicos encontrados no mercado voltados para as mais diferentes funções. As três escolhas para esta categoria se encontram disponíveis na Tabela 52, onde são realizadas comparações que atendem aos requisitos citados anteriormente.

Tabela 52 – Modelos de Interfaces Microcontroladas.

INTERFACE MICROCONTROLADA			
PARÂMETROS	ARDUINO UNO	KIT PIC	RASPBERRY PI 3
Fabricante Microcontrolador	ATMEL	MICROCHIP	CONSÓRCIO
Modelo	ATmega328	Pic18f4550	Cortex-A7
Temperatura de operação	(-40 a 80 °C)	(-40 a 125 °C)	(-10 a 50 °C)
Porta de Alimentação	USB / Externa	USB / Externa	4 x USB
Tensão de Operação	5V	5V	5V
Tensão de Alimentação (recomendada)	7-12V CC	6-15V CC	4,75-5,25V CC
Pinos	14	33	40
Memória Flash	32 KB	32 KB	Expansível
Memória SRAM	2 KB	2 KB	1 GB
EEPROM	1 KB	256 B	-
Sistema Operacional	-	-	Windows, Linux
Linguagem de Programação	C++	C	Todas
Dimensões	68,6 x 53,4 x 14 mm	90 x 45 x 15 mm	85 x 56 x 17 mm
Peso	25 g	32 g	42 g
Preço	R\$ 39,00 - 50,00	R\$ 92,00	R\$ 152,00 - 210,00



Fonte Própria.

Requisitos importantes estão ligados a análise da Interface Microcontrolada, ex-

postos na Seção 4.1.1, portanto algumas considerações são necessárias antes da escolha do melhor modelo para compor a interface microcontrolada deste estudo. Em observação às fichas técnicas dos kits de desenvolvimento apresentados na Tabela 52, observa-se uma evolução em termos de capacidade de memória e processamento principalmente, isto se dá, pelo fato dos modelos Arduino UNO e o Kit Pic serem consideradas placas de prototipagem eletrônica, muito utilizadas para integração com módulos e componentes externos, como *shields*, sensores entre outros, e devido à sua simplicidade de programação e compatibilidade com diversos periféricos vem ganhando espaço entre os desenvolvedores. Em contrapartida, o modelo Raspberry Pi 3, apresenta melhor desempenho técnico pelo fato de sua arquitetura permitir uma melhor experiência em termos de desenvolvimento de *software*, exigindo um maior desempenho em processamento de dados.

Em relação às necessidades de integração de diferentes módulos e componentes eletrônicos, se considera como exigência técnica no âmbito da interface microcontrolada a escolha de um modelo que integre prototipagem rápida, linguagem de programação e que tenha disponível para aquisição dentro de sua comunidade de desenvolvedores módulos prontos, que incorporem aspectos relacionados a comunicação, armazenamento de dados, rastreamento, controle, entre outros, afim de permitir maior agilidade e interação no processo de desenvolvimento do sistema. Considerando estes aspectos o modelo Arduino UNO, possui maior destaque em relação ao Kit PIC, além de sua popularidade em aplicações voltadas para modelos de prototipagem rápida. Outros fatores influenciam nesta escolha, como preço, peso e dimensões, apesar de ambos os modelos atenderem a estas duas últimas.

Entretanto fatores relacionados a desempenho, podem ser limitantes no caso do Arduino UNO, devido a sua quantidade de pinos, reduzir as possibilidades de integração com vários componentes eletrônicos, onde para contornar esta limitação módulos expansivos serão aplicados ao *hardware* com o intuito de permitir atender as funcionalidades exigidas pelo sistema, as quais englobam questões relacionadas a IHM e ao processo de armazenamento de dados, apresentadas ainda nesta seção na definição dos componentes eletrônicos.

4.2.2.2 Radiotransmissor

A etapa seguinte foi a escolha do componente físico do tipo radiotransmissor, onde a princípio buscou-se equipamentos compatíveis com a interface e que atenda a parâmetros regulatórios de frequência e potência de transmissão, requisitos primordiais para o atendimento a legislação vigente, conforme exposto na Seção 2.3.4, Tabela 3 e presentes na lista de requisitos funcionais da Tabela 13.

Durante o processo de exploração dos equipamentos relacionados a transmissão de dados via rádio (radiotransmissor) algumas características como: miniaturização, baixo

custo, frequência de operação, potência de operação, baixo consumo de energia, protocolo de comunicação, otimização de portas e compatibilidade com outras interfaces, motivaram esta etapa, jogando luz encima de tecnologias oriundas da Internet das coisas (IoT, do Inglês *Internet of Things*), cujo conceito está ligado à interconexão digital de objetos cotidianos com a Internet, ou através de redes locais (LAN) ou ponto-a-ponto (P2P), sendo que a conexão de dispositivos à rede através de sinais de rádio de baixa potência tem sido um campo de grande crescimento dentro deste conceito. A partir desta análise buscou-se modelos que estão inseridos neste enfoque e se enquadrem às exigências do sistema em desenvolvimento. A Tabela 53, apresenta três opções para esta categoria, as quais são comparadas para análise e definição da melhor opção.

Tabela 53 – Modelos de Radiotransmissores.

PARÂMETROS	RADIOTRANSMISSOR		
	HUAWEI	EBYTE	UBLOX
Modelo	eLTE-IoT eM300-8a	E32 433T30D	VERA-P173
Frequência Operacional	902 a 928 MHz	410 a 441 MHz	5,9 GHz
Potência de Transmissão Máxima	20dBm	30dBm	23dBm
Baud Rate	9600-57600	1200-115200	-
Sensibilidade	(-136 dBm)	(-145 a -148 dBm)	(-98 dBm)
Tensão de Operação	3,8V	3,3-5,2V CC	3,5-5,0V CC
Temperatura de Operação	(-40 a 75°C)	(-40 a 85°C)	(-40 a 95°C)
Distância (Área Urbana)	5 Km	8 Km	1 Km
Antena	Integrada	SMA-K	Integrada
Biblioteca Arduino	SIM	SIM	NÃO
Dimensões	19,9 x 23,6 x 2,2 mm	24 x 43 x 2,8 mm	24,8 x 29,6 x 4 mm
Peso	1,6 g	8,2 g	2,4 g
Preço	R\$ 150,00	R\$120,00 - 200,00	R\$ 80,00



Fonte Própria.

A escolha pelos modelos presentes na Tabela 53, permitiu um comparativo de âmbito regulatório no contexto da faixa de frequência operacional, onde os equipamentos da HUAWEI e EByte estão inseridos na faixa de UHF, e o da UBLOX na faixa de Banda C, após análise das faixas destinadas ao Serviço de Radioamador por satélite presente na Tabela 3, pode-se observar que apenas o modelo da EByte opera em frequência da faixa UHF autorizada no Brasil, que corresponde ao espectro de 430 a 440 MHz, estas condições são de grande relevância, pois ao adotar equipamentos produzidos em outros países, os mesmos devem ser compatíveis com a legislação regulamentada no país de origem em que

o sistema entrará em operação.

Em relação à potência de transmissão, todos os modelos se enquadraram, sendo que o modelo da EByte proporciona uma potência máxima de até 1W, correspondente a 30 dBm, o que conseqüentemente influencia no alcance de transmissão. Todos os fabricantes apresentam distâncias que correspondem ao uso dentro de áreas urbanas, contribuindo para interferências e conseqüente perda de sinal, o que pode levar o alcance nominal determinado pelo fabricante a conseguir taxas menores do que as informadas. Como o cenário deste estudo está relacionado a transmissão de dados no espaço livre, com visada direta, conforme explanado na Seção 2.3.3, as taxas informadas pelo fabricante podem ser atingidas e ainda serem superadas, o que pode ser melhorado também com a adoção de antenas com ganho diretivo mais alto, no caso do modelo da EByte esta possibilidade de modificação do tipo da antena existe, para os demais modelos ela é inexistente, pelo fato de possuírem antenas integradas ao seu circuito, inviabilizando este tipo de aprimoramento.

Em síntese para o componente físico do tipo radiotransmissor, o modelo da EByte E32 422T30D, atende as especificações regulatórias e ainda possui condições de alcance de transmissão mais otimistas, com possibilidades de possíveis ampliações conforme os padrões de transmissão e características da antena aplicada, que será detalhado em breve.

4.2.2.3 Componentes eletrônicos

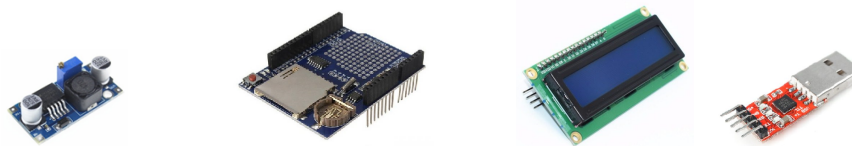
Em continuidade às escolhas dos equipamentos físicos, buscou-se definir os módulos para IHM e armazenamento de dados, sendo que os mesmos foram escolhidos diretamente por ser tratar de funcionalidades específicas as quais, para adoção na Interface Microcontrolada do Arduino UNO são conhecidas como *Shields* ou módulos, ou seja, tratam-se de equipamentos eletrônicos prontos voltados para uso em prototipagem eletrônica para arquiteturas Arduino. Os fabricantes listados nesta categoria, se encontram presentes nos componentes integrados a cada circuito e não se faz necessário sua comparação, já que está sendo considerado o módulo como um todo e não apenas as suas partes em separado. A Tabela 54 desta etapa se preocupa em apresentar requisitos como peso, dimensões e demais condições de operação que agregam informação considerável para a composição do componente Bateria que será avaliado em seguida.

Cada um dos componentes da Tabela 54, tem funções específicas dentro do contexto do sistema modelado, o regulador de tensão tem a funcionalidade de estabilizar a tensão fornecida pela fonte de energia em corrente contínua, ou seja, fontes provenientes de baterias e placas solares, esta regulação se faz necessária afim de evitar oscilações que possam atrapalhar o desempenho operacional dos componentes instalados, garantindo confiabilidade e precisão, requisitos presentes e que foram especificados nas Tabelas 21 e 22.

Os componentes Data Logger e Display LCD, garantem respectivamente a análise

Tabela 54 – Modelos de Componentes Eletrônicos.

PARÂMETROS	COMPONENTES ELETRÔNICOS			
	REGULADOR DE TENSÃO	ARMAZENAMENTO DE DADOS	VISOR LCD	CONVERSOR
Modelo	LM2596	Data Logger Shield	Display 16x2 + I2C	USB TTL
Tensão de Operação	3,2-40V CC	3,0-3,6V CC	5,0V CC	3,0-3,6V CC
Temperatura de Operação	(-40 a 85°C)	(-40 a 85°C)	(-40 a 85°C)	(-55 a 150°C)
Biblioteca Arduino	NÃO	SIM	SIM	NÃO
Dimensões	46 x 22 mm	68 x 53 x 23 mm	80 x 35 x 11 mm	21 x 15,3 x 4 mm
Peso	1,8 g	20 g	50 g	1,5 g
Preço	R\$ 10,00 a 20,00	R\$ 23,00 a 42,00	R\$ 25,00 a 50,00	R\$ 15,00 a 25,00



Fonte Própria.

do comportamento do sistema posterior e em tempo real durante a missão. O Data Logger, presença essencial na EE, preenche os requisitos de armazenamento de dados em cartão de memória e monitoração das atividades realizadas, pacotes transmitidos, pacotes entregues, tempo de transmissão esgotado, entre outras ações que são tratadas durante a operação, para análises posteriores, contribuindo para o atendimento dos requisitos apresentados nas Tabelas 24, 25, 26 e 27

O Display LCD parte importante na ET, atende aos requisitos de monitoração em tempo real, discutido na Tabela 34, permitindo ao usuário o acompanhamento dos status das transmissões, por meio do visor disponível, que situam o usuário em relação ao andamento da operação, para que este possa intervir de imediato na correção de possíveis falhas, que podem contemplar desde ajustes na posição da antena da ET para atender a diretividade, ou mesmo envio de comandos para estabelecer condições de anomalia na operação, em atendimento ao requisito da Tabela 21. Um ponto de destaque no Display LCD é a integração do módulo I2C, protocolo que minimiza a quantidade de portas utilizadas para comunicação com a Interface Microcontrolada, minimizando a quantidade de fios e permitindo a conexão de outros dispositivos nas portas livres, como sensores ou demais componentes que possuem funções específicas, garantindo as recomendações do requisito da Tabela 35.

Por último o conversor USB TTL, tem como função a integração ao radiotransmissor permitindo possíveis parametrizações pelo usuário através do notebook ou computador de mesa. Estes parâmetros constituem valores pré determinados para o correto funcionamento do radiotransmissor, como por exemplo, frequência de operação, potência de transmissão, canal de comunicação, *baud rate*, ou ainda modos de operação do equipamento, estas condições podem ser visualizadas no *datasheet* do módulo, disponível no Anexo B.

4.2.2.4 Baterias

O requisito flexibilização presente na Tabela 23, dita sobre a exigência de fontes de energia alternativas, gerada a partir de baterias ou placas solares, afim de suprir a demanda de energia elétrica proveniente dos equipamentos instalados, para a proposta do cenário estudado este aspecto é atendido por baterias, como fonte principal de geração de energia. A Tabela 55 apresenta três modelos de fabricantes distintos, visando o fornecimento em concordância com a operação do sistema.

Tabela 55 – Modelos de Baterias.

BATERIAS			
PARÂMETROS	RONTEK	ZIPPY	TUNIGY
Modelo	SC	63380	2200mah 2s
Material	Níquel e Cádmio	Lítio e Polímeros	Lítio e Polímeros
Tensão	7,2V	7,4V	7,4V
Corrente	1800 mAh	1500 mAh	2200 mah
Conector	Tamiya macho	XT60	XT60
Dimensões	22,7 x 130,3 x 45 mm	24 x 107 x 13 mm	104 x 35 x 16 mm
Peso	0,350 g	99 g	120 g
Preço	R\$ 85,00 - 120,00	R\$ 180,00 - 210,00	R\$ 100,00 - 150,00



Fonte Própria.

As baterias são os elementos integrados a parte interna do sistema que possuem peso e dimensões consideráveis, analisando o espaço do CubeSat que tem medidas padronizadas de 10 cm de arestas, conforme relatado na Seção 2.2, deve-se pensar nestes componentes, em especial para a EE, com medidas inferiores a estrutura do satélite. No caso das baterias apresentadas, nenhuma se enquadrou às medidas, desta forma, para o embarque dos modelos de baterias apresentados, a sua posição dentro da estação CubeSat ocupará mais espaço devido o seu posicionamento irregular. Entretanto, o tamanho das baterias dos tipos analisadas está diretamente ligado a sua capacidade de fornecimento de energia, sendo assim, baterias com fornecimento de tensão menor que as listadas na Tabela 55, possuem uma autonomia menor, que pode prejudicar o funcionamento do sistema como todo, deixando de suprir a energia necessária para a realização de uma missão por completo, em consequência dispositivos podem funcionar incorretamente e apresentar anomalias nos resultados fornecidos.

No contexto desta seção o modelo escolhido foi o da fabricante Rontek, primeiro

devido o custo e segundo pela facilidade na aquisição no mercado local, em contrapartida o material químico que compõe a sua estrutura é inferior as dos demais fabricantes, com tempo de vida útil menor, além de possuir dimensões elevadas em relação as demais. A escolha neste caso não se deu pelo custo-benefício e nem pelos comparativos aos demais parâmetros, mas em conformidade ao que foi exposto no início deste parágrafo, valendo uma reanálise deste quesito em uma próxima iteração.

4.2.3 Mapa tecnológico - Domínio mecânica

Nesta seção o posicionamento detalha o âmbito do domínio mecânico integrado ao sistema. A Figura 27 apresenta os respectivos fabricantes para componentes do tipo Antenas, separando estas de acordo com a sua forma de radiação do sinal.

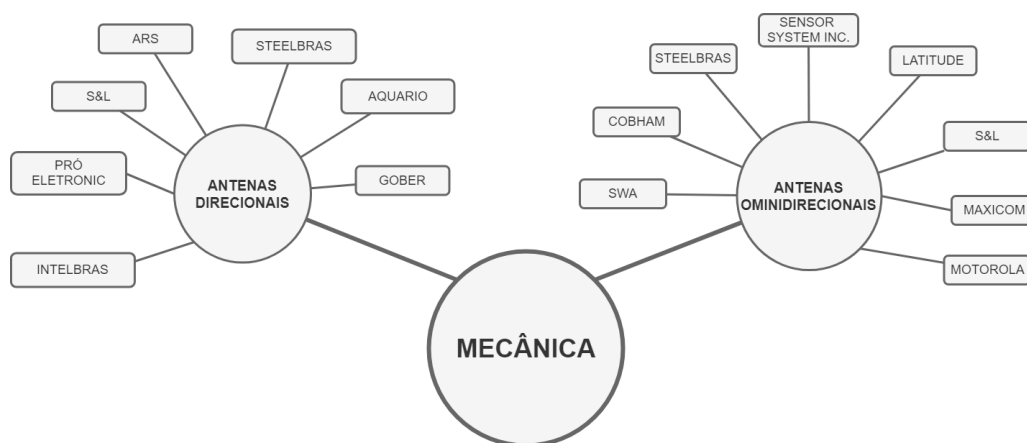


Figura 27 – Mapa de fabricantes - Domínio mecânica.
Fonte Própria.

O mapa tecnológico do domínio mecânica, apresentado na Figura 27, listou os diversos fabricantes para as antenas dos tipos direcionais e omnidirecionais, devido a ampla gama de fabricantes, a pesquisa estará limitada a apenas três modelos, escolhidos conforme requisitos levantados considerando condições como: faixa de operação de frequência (Tabela 14), potência de operação (Tabela 15), peso (Tabela 20), dimensões (Tabela 19), feixe diretivo (Tabela 28), ganho (Tabela 32), além de serem facilmente encontrados no mercado local e com preços acessíveis (Tabela 16), visando a implementação do protótipo futuro, após as fases de testes e verificações do modelo virtual.

A escolha das antenas esta diretamente relacionada a faixa de frequência de operação e potência de transmissão, as quais foram definidas durante a definição do modelo do Radiotransmissor na Seção 53, as especificações definidas nesta seção ditaram que o sistema deverá trabalhar na faixa de frequência de UHF, mais precisamente dentro do espectro compreendido entre 430 a 440 MHz, destinado ao Serviço de Radioamador via satélite. Em atendimento a esta exigência o modelo E32 433T30D da fabricante EByte foi o escolhido,

por trabalhar dentro da faixa regulamentada, com limites de operação de 410 a 441 MHz e limites de potência de transmissão até 1W. Portanto estas características devem estar presentes nas especificações das antenas que integrarão a arquitetura das respectivas estações.

Antes do mapeamento dos fabricantes correspondente a cada tipo de antena, faz-se necessário a conceituação dos dois tipos de antenas presentes na Figura 27, quanto a sua forma de radiação do sinal. As antenas possuem características de radiação distintas, podendo ser omnidirecionais e direcionais (ou diretivas). As antenas omnidirecionais possuem um diagrama de radiação uniforme, irradiando para todas as direções com o mesmo ganho. São as mais simples de se fabricar, mas oferecem baixo alcance ao sinal. Já as antenas direcionais concentram a maior parte da energia irradiada em uma determinada direção em detrimento das demais, aumentando o seu ganho. As seções a seguir apresentam cada uma destas e seus respectivos modelos levantados. As características de propagação do sinal em relação à sua polarização são apresentadas na Figura 28 para antenas omnidirecionais, e na Figura 29 para antenas direcionais.

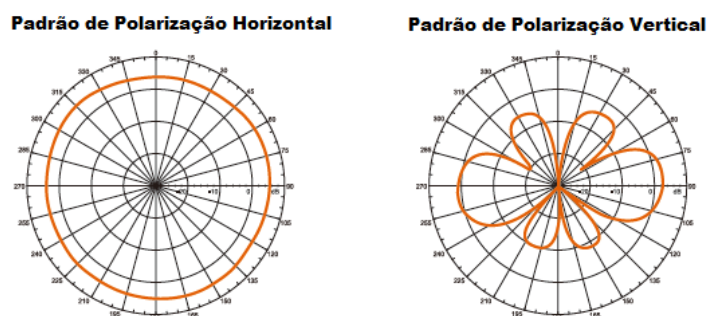


Figura 28 – Padrão de radiação do sinal - Antena omnidirecional.
Fonte: TP-LINK (2017).

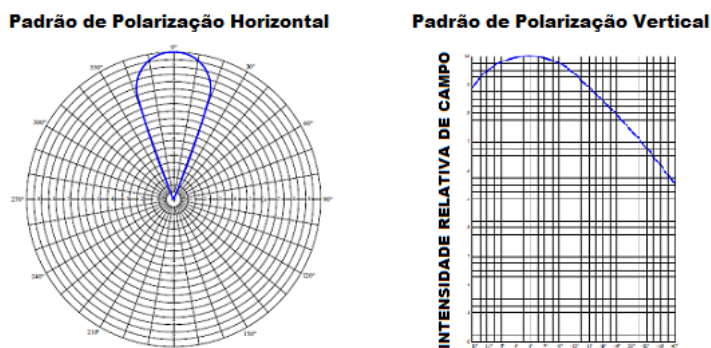


Figura 29 – Padrão de radiação do sinal - Antena direcional.
Fonte: Gober (2018).

Inserido no âmbito do domínio mecânico, o sistema apresenta um desafio para composição da integração do domínio eletrônica para o cenário da ET. Esta necessidade vem

de encontro a elaboração de uma unidade de condicionamento funcional de componentes, ou seja, uma unidade integrada das interfaces eletrônicas, permitindo sua comunicação com o ambiente externo e proporcionando condições mínimas de operação ao usuário, considerando aspectos de segurança, mobilidade, portabilidade e compatibilidade.

Esta unidade segue a tendência COTS, em que a aquisição da proposta de solução entra em concordância com os requisitos funcionais: dimensões (Tabela 19), peso (Tabela 20), controlabilidade (Tabela 21), precisão (Tabela 22), flexibilização (Tabela 23), exatidão (Tabela 29), monitoração (Tabela 34), compatibilidade de conexão (Tabela 35), versatilidade (Tabela 36).

4.2.3.1 Antenas omnidirecionais

As antenas omnidirecionais tem um padrão de radiação similar, fornecendo uma radiação horizontal de 360 graus. Elas são usadas quando a cobertura é exigida em todas as direções (horizontalmente) da antena com graus de variação de cobertura vertical. A polarização é a orientação física do elemento na antena que emite realmente a energia de RF. Uma antena omnidirecional, por exemplo, é geralmente uma antena polarizada vertical. Estas características fazem deste tipo de antena, atraente para integrar a EE, pois como não existem controle da posição da antena durante a missão, seu espectro de radiação em 360° compensa esta deficiência.

A Tabela 56 apresenta três modelos de fabricantes de antenas, cujos produtos apresentam a característica de radiação do tipo omnidirecional.

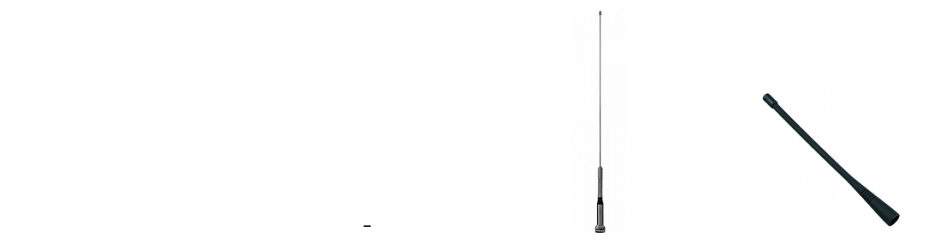
Dentre os modelos, a marca S&L não forneceu imagens do seu produto, sendo esta a que apresentou melhor ganho, o que traz benefícios em relação ao alcance durante a operação. As demais apresentam características similares com grandes diferenças em relação a peso e dimensões. Considerando para esta etapa os requisitos de peso (Tabela 20), dimensões (Tabela 19), pelo equipamento ser parte integrante da EE, além de custo (Tabela 16) e facilidade de aquisição no mercado local, a escolha recaiu sobre o modelo da MOTOROLA, ATU-6B, a preferência também pesou por esta marca esta diretamente ligada a produção de periféricos para rádios de comunicação sem fio, o que favorece em termos de expertise na fabricação de equipamentos relacionados a área de radioamadorismo.

4.2.3.2 Antenas direcionais

As antenas direcionais são equipamentos que tem a propriedade de enviar ou receber ondas eletromagnéticas com mais eficiência em algumas direções do que em outras. A diretividade máxima dessa antena é significativamente maior do que a de uma antena do tipo omnidirecional. Possuem características de radiação que levam à concentração de potência radiada numa determinada direção do espaço, estas características são: alta diretividade ou ganho, feixe de meia potência estreito e alta relação frente-costas. A

Tabela 56 – Modelos de Antenas omnidirecionais.

ANTENAS OMNIDIRECIONAIS			
PARÂMETROS	S&L	STEELBRAS	MOTOROLA
Modelo	SLMU-3DMAGT	AP6200	ATU-6B
Frequência	420 a 450 MHz	430 a 470 MHz	420 a 450 MHz
Espectro	UHF	UHF	UHF
Ganho	5,15 dBi	3 dBi	3 dBi
Potência Máxima	100 W	100 W	100 W
Polarização	Vertical	Vertical	Vertical
Impedância	50 Ohms	50 Ohms	50 Ohms
Conector	UHF	UHF	SMA-Macho
Dimensões	500 mm	575 mm	155 mm
Peso	370 g	136 g	60 g
Preço	R\$ 90,00 - 160,00	R\$ 120,00 - 170,00	R\$ 30,00 - 70,00



Fonte Própria

sua escolha para compor o sistema da ET, vem de encontro a estas necessidades, o que amplia o seu alcance de transmissão em relação a EE, atendendo ao requisito funcional de capacidade de alcance do feixe direcional, apresentado na Tabela 28.

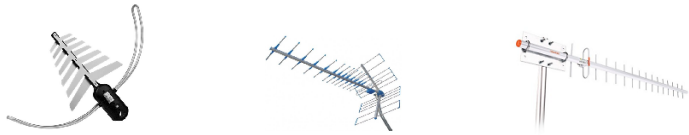
A Tabela 57 apresenta três modelos de fabricantes de antenas, cujos produtos apresentam a característica de radiação do tipo direcional.

Os modelos apresentados são específicos para aplicação no sistema ET, portanto a questão de peso e dimensões reduzidos neste escopo não se faz necessária, os requisitos importantes neste cenário, diz respeito a faixa de frequência de operação, conforme Tabela 14; potência de transmissão, Tabela 15; precisão, Tabela 22; capacidade de alcance do feixe direcional, Tabela 28; garantia de operação, Tabela 32 e compatibilidade de conexão, Tabela 35, estes fatores são influenciados diretamente pela escolha da antena direcional, pois quanto melhor a sua diretividade, mais garantias na transmissão eficiente do sinal.

A escolha do melhor modelo fica para o fabricante Gober, GY450, com especificações inseridas nas faixas de transmissão exigidas pelo radiotransmissor, além do atendimento às normas técnicas de compatibilidade de polarização e impedância, esta antena apresenta um aspecto bastante importante relacionado a sua frequência de operação, trabalhando

Tabela 57 – Modelos de Antenas direcionais.

ANTENAS DIRECIONAIS			
PARÂMETROS	AQUARIO	GOBER	AQUARIO
Modelo	DTV-3200	GY450	CF-917
Frequência	54 a 806 MHz	430 a 470 MHz	890 a 960 MHz
Espectro	VHF/UHF/FM	UHF	UHF
Tipo	Log Periódica	Yagi	Yagi
Ganho	6 Dbi	10 dBi	17 dBi
Polarização	Horizontal	Horizontal/Vertical	Horizontal/Vertical
Impedância	75 Ohms	50 ou 75 Ohms	50 Ohms
Conector	N-fêmea	N-fêmea p/ 50 Ohms	N-fêmea
Dimensões	715 x 510 x 85 mm	1030 x 350 mm	160 mm
Peso	555 g	2,10 Kg	1,58 kg
Preço	R\$ 190,00 - 240,00	R\$ 623,00 - 685,30	R\$90,00 - 150,00



Fonte Própria.

em uma faixa bastante estreita de 430 a 470 MHz, dentro do espectro de UHF, propicia condições operacionais com reduzido impacto de interferências oriundas de outras fontes de transmissão, seu diagrama de radiação e demais especificações técnicas, estão disponíveis no Anexo C, página 255. Em relação a conexão com o radiotransmissor se faz necessário a aquisição de um conector do tipo PigTail em ambos os modelos para a devida compatibilidade dos encaixes.

Apesar do preço elevado em relação às demais opções, os benefícios em termos de transmissão e recepção influenciam bastante no alcance de uma maior precisão e acertos durante as missões, esses requisitos podem ser ampliados com uma combinação dos componentes radiotransmissor e antena em sintonia com as especificações técnicas exigidas.

4.2.3.3 Unidade de Condicionamento Funcional

A composição da Unidade de Condicionamento Funcional, propõe embarcar as soluções do domínio eletrônica em uma única unidade para integração da ET, representada por uma caixa protetora que ofereça condições de proteção aos componentes em relação as intempéries externas, neste caso buscou-se como solução um modelo de caixa hermética que minimiza a entrada de poeira e umidade.

A partir da solução proposta, fez-se necessário buscar condições para a caixa que

atendam a requisitos de dimensões e peso impostos pelos componentes que serão integrados, desta forma a Tabela 58, apresenta apenas um modelo para esta solução, pois a mesma não tem a necessidade de seguir padrões específicos, sendo necessário atentar apenas aos descritivos apresentados.

Tabela 58 – Modelo Caixa hermética.

CAIXA HERMÉTICA	
PARÂMETROS	DESCRIÇÃO
Proteção	Grau IP 65
Eficiência	Anel tipo oring
Abertura	Tampa superior com dobradiça e fecho duplo
Dimensões	200 x 200 x 110 mm
Preço	R\$ 30,00 - 60,00



Fonte Própria.

4.2.4 Arquitetura lógica

A arquitetura do subsistema de comunicação no que tange o domínio lógico, foi especificada nos requisitos não-funcionais em concordância com as características que vinculam as condições de operação das estações ET e EE ao domínio cruzado específico, correspondente ao desenvolvimento de *software*, compostos pelos tipos de funções lógicas, definidas a partir da sua interação com os requisitos funcionais e seus respectivos cenários físicos.

Com o objetivo de proporcionar um esclarecimento em relação as funções citadas durante a etapa de levantamento de requisitos, busca-se através de uma notação simples, ilustrada pelo diagrama de Caso de Uso da Figura 30, apresentar a interação entre o sistema representado pelas estações ET e EE e suas atividades específicas, distinguidas inicialmente pelas ações macro, realizadas por cada agente envolvido.

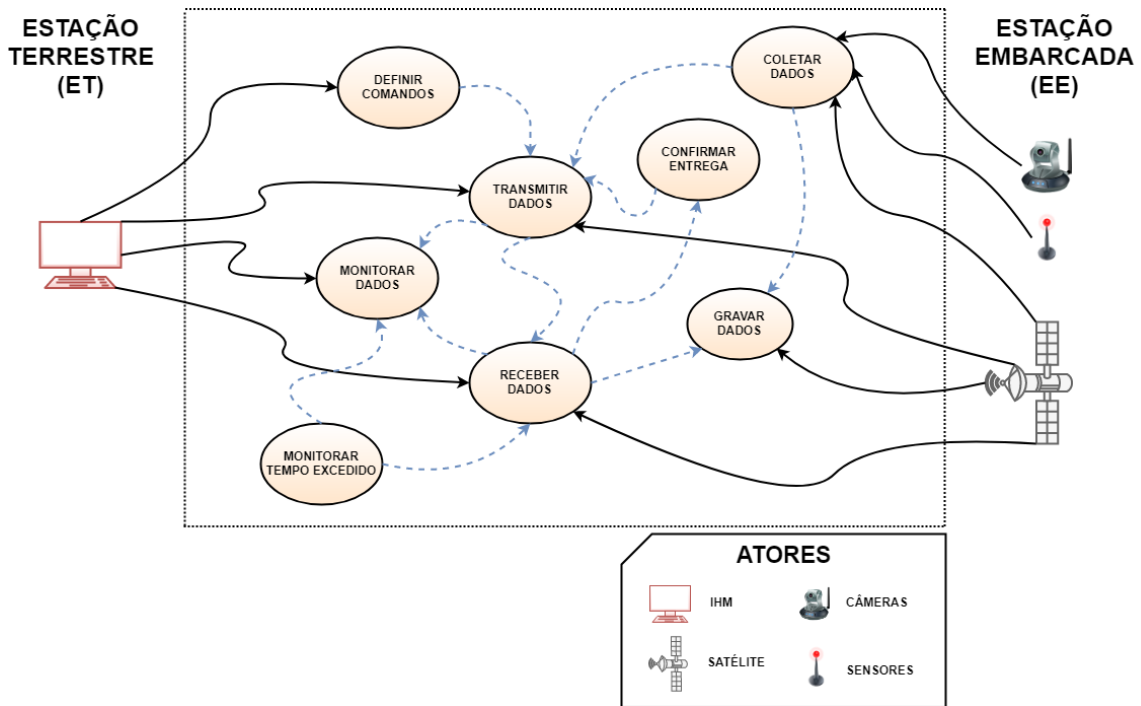


Figura 30 – Diagrama de caso de uso.
Fonte Própria.

Cada uma das atividades presentes na Figura 30, definem as expectativas em relação ao sistema de cada estação, e as operações executadas durante a missão, onde as ações principais "Transmitir Dados" e "Receber Dados" são comuns no âmbito geral. Em contrapartida as atividades de comando, "Definir Comando", e telemetria, "Coletar Dados", representam respectivamente o *uplink* da ET para EE e o *downlink*, considerando o fluxo inverso, simbolizando assim a sigla T&C do subsistema de comunicação. Conforme mencionado anteriormente, a intervenção de comando é realizada pelo operador por intermédio de uma IHM dentro de parâmetros específicos destinados ao envio de mensagens, acionamento, controle ou desligamento de determinados componentes embarcados, enquanto que a telemetria fornece dados coletados a partir de sensores, imagens provenientes de câmeras ou comportamentos diversos referentes a saúde do satélite.

Outras atividades específicas executadas internamente pelo sistema, são "Monitorar Tempo Excedido" e "Confirmar Entrega". O primeiro representa o *Time Out*, ou seja, tempo de espera em que o sistema aguarda a confirmação com sucesso do pacote transmitido. O segundo é exatamente a confirmação da entrega do pacote, sendo executada na EE. Estas ações caminham juntas, promovendo uma cooperação mútua no objetivo final da correta transmissão e recepção dos dados.

Em relação a implementação da interface homem-máquina, do lado da ET está a atividade "Monitorar Dados", simbolizada pela entrega das informações de forma compreensível ao entendimento humano, através de um monitor ou visor. E do lado da EE, se encontra a atividade "Gravar Dados", vinculada ao banco de dados, composto por um *log*

A Figura 31, compõe o *workflow* de trabalho para cada estação, por meio de atividades que se interrelacionam, apresentando tomadas de decisões e condições de desvios condicionais que permitem entender a integração na comunicação da ET para EE e vice-versa. As caixas retangulares presentes no contexto do diagrama, representam as funções citadas durante a etapa de mapeamento dos requisitos e estão listadas da seguinte forma:

1. Estação Terrestre (ET):

- IHM - Função Display LCD;
- Transmite Dados - Função Transmissão;
- Recebe Dados - Função Recepção;
- Excede Tempo - Função TimeOut.

2. Estação Embarcada (EE):

- Recebe Dados - Função Recepção;
- Confirma Entrega - Função Confirmação de Entrega;
- Transmite Dados - Função Transmissão;
- Grava Dados - Função Log;
- Coleta Dados - Função Coleta de Dados.

Em análise às funções, nota-se que as atividades de transmissão e recepção de dados, estão presentes em ambas estações, devido a estrutura física para o *hardware* ser a mesma, logo estas funções compartilham da mesma estrutura em seu trecho de código para as Funções de Transmissão e Recepção. Quanto as demais, cada uma realiza papel específico dentro da operação computacional, em atendimento a modelagem e ao ambiente em que se encontra relacionado, esse *template* segue um padrão de arquitetura.

Segundo Pressman (2002), como todas as técnicas de modelagem usadas na engenharia de sistemas e de *software*, o padrão da arquitetura possibilita que o analista crie uma hierarquia de detalhes, estabelecendo uma fronteira de informação entre o sistema e o ambiente que o sistema vai operar. De posse deste conceito, um modelo de arquitetura é representado na Figura 32, onde são atribuídos os elementos de sistemas das regiões de processamento que envolvem o subsistema de comunicação, representando um modelo adaptado para as condições do cenário em estudo.

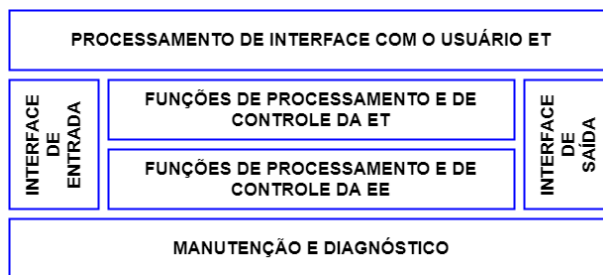


Figura 32 – Modelo de arquitetura lógica.
Fonte Própria.

O modelo da Figura 32, destaca cinco regiões de processamento:

- (a) interface com o usuário;
- (b) entrada;
- (c) função e controle do sistema;
- (d) saída;
- (e) manutenção e diagnóstico.

Por meio destas regiões, pode-se definir o ADC (do Inglês, *Architecture Context Diagram*), destacando os produtores externos, fornecedores de informações utilizadas pelo sistema, todos os consumidores externos de informação adotada pelo sistema e todas as entidades que se comunicam por meio da interface ou realizam manutenção e diagnóstico. Para ilustrar este processo a Figura 33, considera uma versão ampliada deste diagrama para o subsistema de comunicação, alvo desta pesquisa.

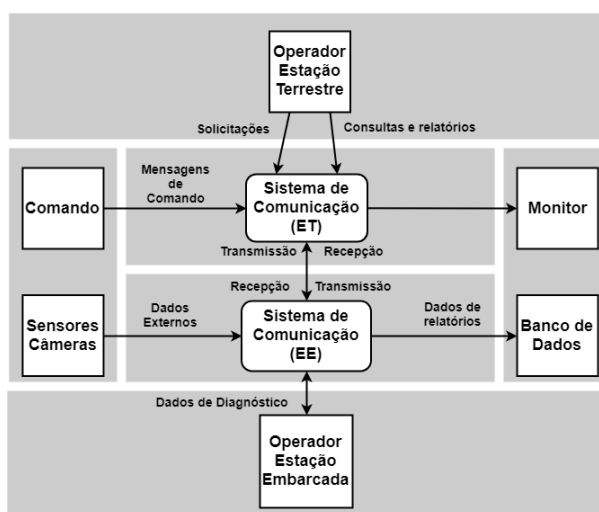


Figura 33 – Diagrama de contexto da arquitetura para o subsistema de comunicação.
Fonte Própria.

Cada retângulo da Figura 33 representa uma entidade externa - ou seja, um produtor ou consumidor de informação do sistema, e são representados pelos seus respectivos elementos físicos, como por exemplo, o Monitor, indica a saída de informações que serão apresentadas ao usuário, no caso deste estudo esta entidade é representada pelo Display de LCD, pertencente ao domínio físico eletrônico. Os relacionamentos das entidades aos seus respectivos agentes físicos serão apresentados na próxima seção, onde serão realizados os vínculos entre os domínios físicos já delimitados neste estudo.

O aprimoramento do ADC, é realizado a partir do detalhamento das funções de processamento e de controle das respectivas estações, neste sentido, são identificados os subsistemas detalhando o Sistema de Comunicação (T&C) e seu funcionamento dentro do contexto esperado. Os subsistemas ou módulos de sistema são definidos em um DFA (do Inglês, *Architecture Flow Diagram*), que é derivado do ADC.

O diagrama de fluxo da arquitetura (DFA) apresentado na Figura 34, mostra os grandes subsistemas e linhas de fluxo de informação (dados e controle) necessários. Além disso, o modelo adotado para a arquitetura na Figura 32, divide o processamento dos subsistemas detalhando os blocos principais hierarquicamente, expandido em novos blocos, conforme a necessidade de detalhamento, no caso deste estudo todos os detalhamentos foram integrados em um único conjunto e são apresentados na Figura 34.

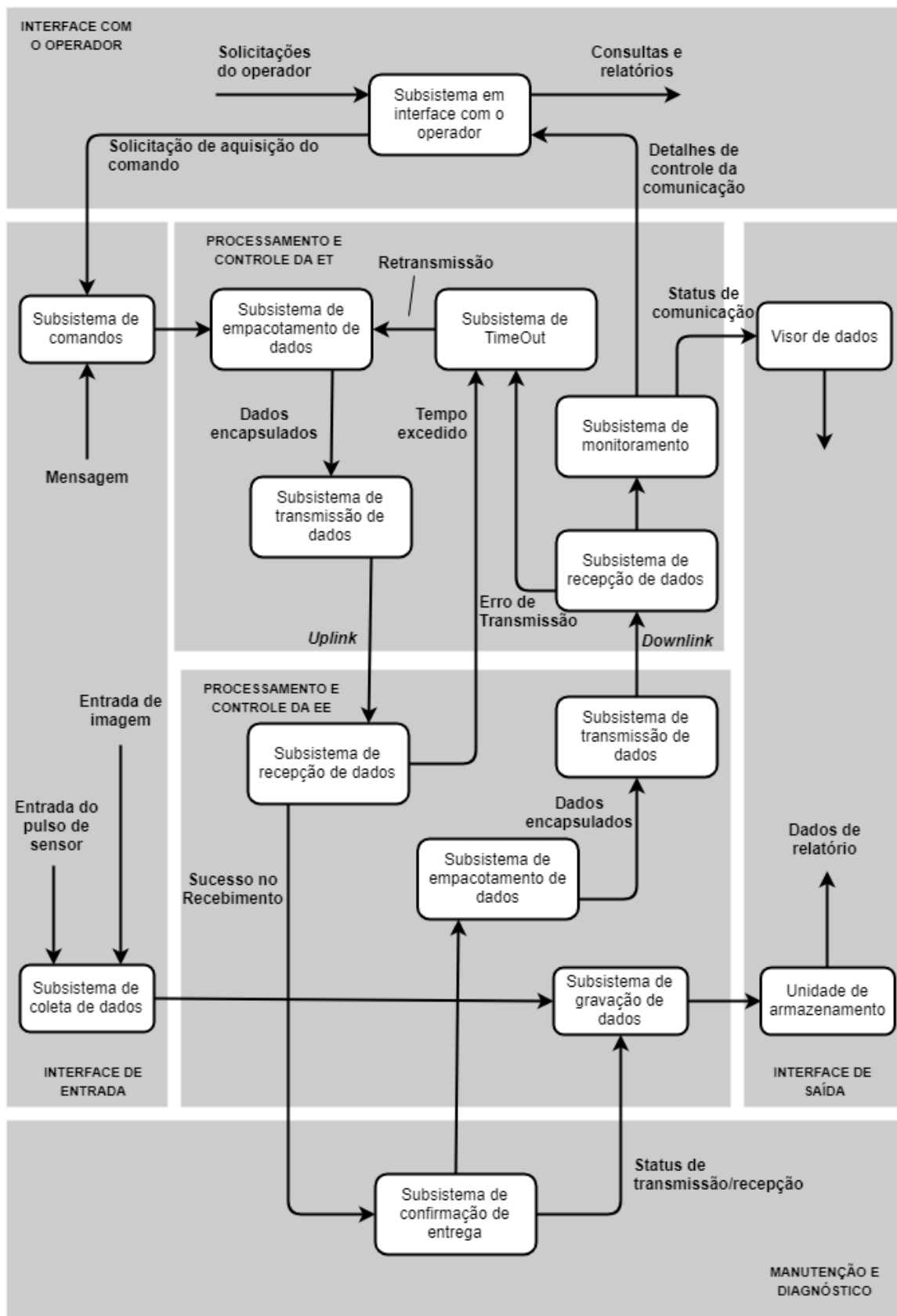


Figura 34 – Diagrama de fluxo da arquitetura para o subsistema de comunicação. Fonte Própria.


4.3 Arquitetura proposta - Integração dos domínios

A arquitetura do sistema é uma representação de um modelo em que existe um mapeamento de funcionalidades para componentes de *hardware* e *software*, um mapeamento da arquitetura de *software* em relação aos componentes físicos (periféricos) e uma interação humana entre esses componentes (IBM, 2006). A partir deste conceito pode-se resumir que a arquitetura de sistema trata-se de:

- a estrutura do sistema em termos dos elementos, componentes e peças;
- os relacionamentos entre esses elementos;
- os requisitos que afetam os elementos e seus relacionamentos;
- o comportamento mostrado pelo sistema e as interações que ocorrem entre os elementos para produzir esse comportamento;
- os princípios, regras e análise racional que caracterizam o sistema (e controlam sua evolução);
- as características e propriedades físicas e lógicas do sistema;
- e por fim a finalidade do sistema.

Em atendimento ao processo de estrutura do sistema em termos de elementos físicos, componentes e peças a Tabela 59 sintetiza as melhores soluções escolhidas para os domínios específicos do sistema, fazendo uma relação as suas respectivas estações.

Tabela 59 – Soluções Domínio Físico

ELETRÔNICA		MODELOS	ET	EE
INTERFACE MICROCONTRALADA			X	X
RADIOTRANSMISSOR			X	X
			X	X
COMPONENTES ELETRÔNICOS				X
			X	
			X	
BATERIA			X	X
MECÂNICA		MODELOS	ET	EE
ANTENA OMNIDIRECIONAL				X
ANTENA DIRECIONAL			X	
CAIXA HERMÉTICA			X	

Fonte Própria.

O motivo de como estes elementos, componentes e peças se relacionam é apresentado na Figura 35, compondo a arquitetura resultante das iterações com os requisitos funcionais, integrada a análise dos pontos de vista dos domínios específicos de *hardware*. Exemplos peso, dimensões, capacidades, modo de operação, entre outros. Neste estudo estes pontos de vista estão representados pelas áreas Eletrônica e Mecânica, as quais foram visualizadas durante a análise das interfaces sob a ótica de uma especificação concreta, o que permitiu a escolha dos equipamentos em resposta as abstrações realizadas na etapa de requisitos.

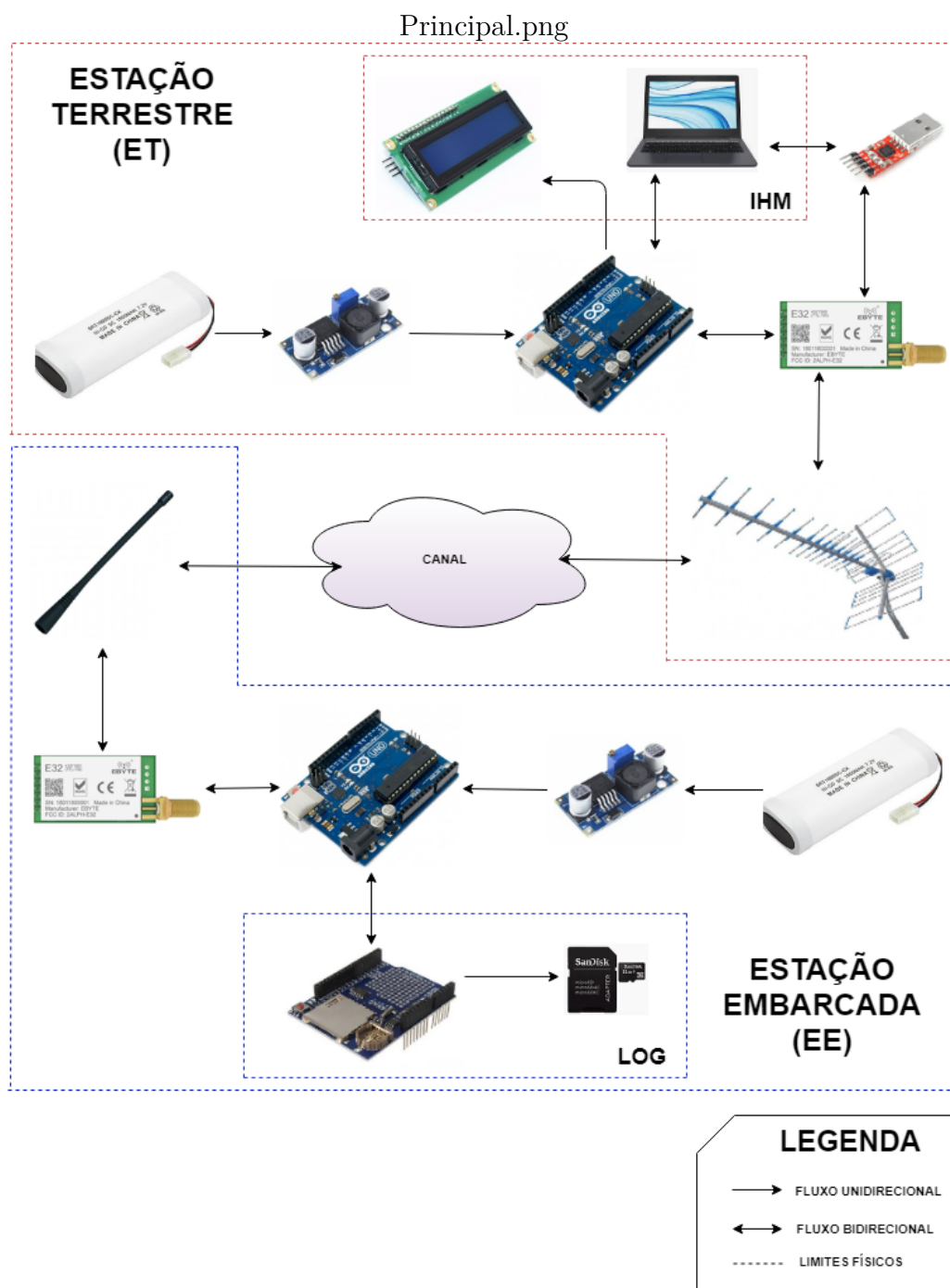


Figura 35 – Arquitetura - Domínio Físico.
Fonte Própria.

A integração dos domínios físicos e lógicos, ocorre na memória de programação, integrada a interface microcontrolada, representada neste estudo pelo modelo ATmega328, cujas especificações técnicas estão disponíveis no Anexo D, página 257. O código de programação referente a arquitetura lógica, é gravado na memória *flash*, do modelo citado, para atender as instruções de operação exigidas pelo sistema, buscando viabilizar o correto funcionamento dos demais componentes físicos embarcados. Este diagrama de integração é apresentado na Figura 36

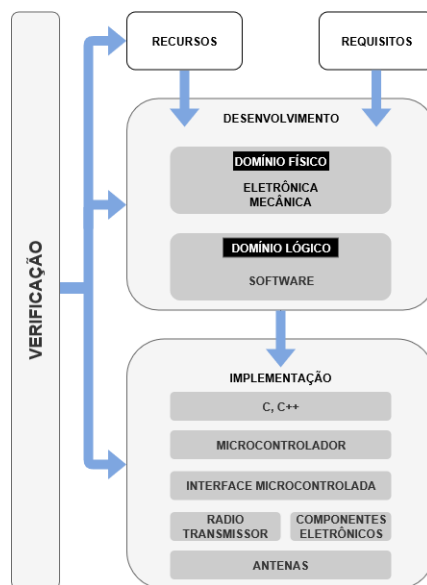


Figura 36 – Arquitetura integrada.
Fonte Própria.

4.4 Resumo do capítulo

O Capítulo 4 apresentou a modelagem do subsistema de comunicação, iniciando pelo levantamento das necessidades dos clientes, que permitiu o rastreamento dos requisitos funcionais, relacionando as prioridades, vínculos e quantificações e dos não-funcionais, vinculados ao comportamento em termos de usabilidade, desempenho, segurança e padrões. Os requisitos foram separados quanto a abrangência de atuação dentro do sistema, relacionando a necessidade ou não destes em cada cenário das estações ET e EE.

Os requisitos possibilitaram o rastreamento das arquiteturas físicas e lógicas, contemplando para a primeira componentes do tipo COTS de baixo custo e na segunda as reais necessidades de operação do sistema para atendimento das exigências inseridas nas respectivas estações, onde a partir da definição da arquitetura para cada domínio específico, pode-se englobar ao sistema como todo.

Com a arquitetura definida o próximo capítulo apresenta os resultados obtidos, partindo da construção do Modelo Virtual, passando pelas técnicas de iteração adotadas pelo MBD: MIL, SIL, e HIL, para validação da integração dos domínios específicos, onde

são gerados os códigos de programação conforme os testes e verificações para validação do sistema, finalizando com as discussões complementares.

5 Testes, Resultados e Discussões

Neste capítulo são apresentados os resultados da integração dos domínios físicos e lógicos simulados através do modelo virtual elaborado, o que permitiu analisar os ensaios de operação do subsistema de comunicação no âmbito de transmissão e recepção de dados no cenário de telemetria (*downlink*) e comando (*uplink*), além da implementação do protótipo físico em atendimento as indicações da metodologia e das necessidades dos usuários.

5.1 Fluxo de testes

A modelagem virtual consiste em uma técnica que faz uso de simulações, com o objetivo de validar um projeto antes que o *hardware* esteja disponível. Nos casos em que a planta e o ambiente de aplicação ainda não são totalmente conhecidos ou compreendidos, como uma construção mecânica, pode ser necessário usar um protótipo de *hardware* para experimentos na construção do modelo. O conhecimento adquirido é, então, armazenado no modelo, permitindo a transferência para outros desenvolvedores, departamentos, fornecedores e clientes.

Seguindo este raciocínio, os resultados deste estudo seguem a sequência de testes apresentada na Figura 37, baseada nas técnicas do MBD, em correspondência às caixas de imersão da Matriz de Adoção apresentada na Tabela 6, para atendimento dos níveis MBD-4 ao MBD-7.

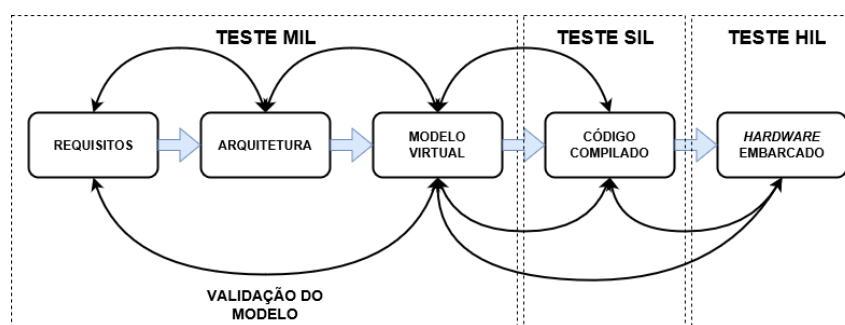


Figura 37 – Sequência de testes.

Fonte Própria.

5.1.1 Modelo Virtual

A elaboração de modelos é um processo iterativo que usa a simulação para transformar um modelo de sistema de baixa fidelidade em uma implementação de alta fidelidade.

Esta etapa permite que todo o sistema seja continuamente testado e integrado aos domínios físicos e lógicos. Quando há um nível adequado de detalhes, as partes que

descrevem o *software* embarcado podem ser usadas na geração de código para testes de prototipagem rápida e *hardware* em *loop* (HIL). Com ainda mais detalhes, o modelo pode ser usado na geração de código de produção.

No desenvolvimento do sistema em estudo, o modelo virtual foi introduzido na etapa de integração do sistema, conforme ilustra a Figura 38.

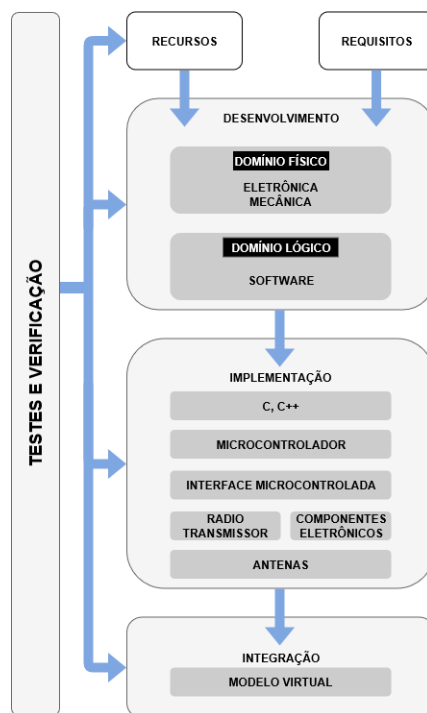


Figura 38 – Integração Modelo Virtual.
Fonte Própria.

A construção do modelo virtual foi considerada nesta etapa, em conformidade ao fluxo do Modelo V apresentado na Figura 21, permitindo a continuidade da etapa de desenvolvimento, com ênfase na integração dos domínios específicos.

O subsistema de telemetria e comando, deste modelo, foi elaborado com o auxílio da ferramenta de modelagem *Simulink*[®], aplicativo incorporado ao *Matlab*[®], que apoiou a construção do diagrama de blocos, possibilitando a simulação das condições de transmissão e recepção desejáveis, inseridas em um cenário próximo da realidade.

Na etapa de levantamento de requisitos, foram definidas as necessidades regentes em conformidade com as exigências do subsistema. Já no contexto dos domínios específicos, se estipulou os modelos físicos e detalhadas as condições de operação correspondentes as regulamentações vigentes, entre outras funcionalidades complementares. No modelo virtual, estas situações foram modeladas em variáveis, sendo nomeadas e valoradas para elaboração do modelo matemático e geração dos resultados simulados, conforme Tabela 60.

Tabela 60 – Modelagem das variáveis.

VARIÁVEIS MODELADAS			
DESCRIÇÃO	REFERÊNCIA	VALOR	UNIDADE
Frequência (Tx/Rx)	sistema_CarrierFreq	433	MHz
Pulso de Entrada (Tx)	sistema_Pot	30	dBm
Amplitude		1	un
Período		125	ms
Largura de pulso	sistema_wf	aleatório	
Ganho Antena Tx	sistema_GainTx	10	dB
Perda no Caminho	sistema_PathLoss	225	dB
Velocidade do Alvo	sistema_targetSpeed	5	m/s
Ganho Antena Rx	sistema_GainRx	3	dB
Potência Tx		1	W
Tempo de Simulação	sistema_SimulationTime	120	s
Distância	sistema_Dist	10	Km

Fonte Própria.

Dentre as informações presentes na Tabela 60, determinadas variáveis possuem valor fixo, definido previamente durante o processo de escolha do componente em seu domínio físico, com destaque para: Frequência, Pulso de entrada (Potência Tx), Ganho Antena (Tx), Ganho Antena (Rx). A variável Perda no Caminho, é calculada com base na Equação (2.7), que considera a frequência e a distância aplicada. A Velocidade do Alvo, corresponde ao deslocamento durante a etapa de subida e descida da plataforma, considerando o transporte realizado por um balão estratosférico e desconsiderando a variação da massa embarcada, estabeleceu-se, neste caso, um valor ideal definido pela equipe do projeto LAICAnSat. As demais variáveis, Tempo de Simulação limitou-se a um valor mínimo para coleta dos resultados, para a Distância buscou-se atender os limites máximos estabelecidos pelo fabricante do componente Radiotransmissor.

Com as variáveis definidas, foram escolhidos os blocos apropriados do *Simulink*[®] disponíveis nos seguintes pacotes:

- *Communications Toolbox*[™], responsável por disponibilizar componentes para análise, design, simulação de ponta-a-ponta e de verificação de sistemas de comunicação;
- *RF Blockset*[™], voltado para projetos de comunicações RF, geração de modelos de transceptores e sistemas de radar;
- *Simulink (Real Time)*[™], para aplicação de blocos que simulam instrumentos de medição.

O modelo considera, como principal requisito, a similaridade das funcionalidades do bloco com os componentes definidos nos domínios da eletrônica e mecânica. O projeto elaborado agrega funções que modelam os recursos essenciais de um sistema de comunicação, mais especificamente, um subsistema de telemetria e comando. Os blocos utilizados incluem um gerador de pulsos (simulação de envio de mensagens), um radiotransmissor Tx, uma antena Tx, uma representação disponível no *Simulink*[®], para o espaço livre, simulando o canal de comunicação e a perda de transmissão no caminho, a representação de um alvo em movimento, uma antena Rx e um radiotransmissor Rx. Os blocos são detalhados a seguir, em atendimento a ordem do fluxo transmissão-recepção, independente da origem da transmissão, podendo ser proveniente da ET ou EE.

5.1.1.1 Bloco gerador de pulsos

O Bloco Gerador de Pulso cria um sinal de frequência com um determinado ciclo de trabalho. O objetivo deste bloco consiste em simular o envio de um sinal proveniente da etapa de telemetria ou comando, correspondendo à transmissão de um evento, oriundo de um teclado ou em resposta a uma ação específica. A Figura 39 ilustra este bloco e suas subfunções.



Figura 39 – Detalhamento do bloco gerador de pulso.
Fonte Própria.

A interface física para representação deste bloco, está ligada ao domínio físico da região IHM, no caso da ET, e ao domínio lógico, *software*, em relação à transmissão da confirmação de entrega para o caso da EE. Em resumo, trata-se dos dados a serem transmitidos.

5.1.1.2 Bloco radiotransmissor Tx

O processo de elaboração do modelo virtual, seguiu uma análise macro do dimensionamento dos blocos, buscando atender às condições presentes na documentação do fabricante (EBYTE, 2017) para o componente Radiotransmissor E32-433T30D adotado neste estudo. As subfunções internas deste bloco, como: modulação, multiplexação, filtragem, amplificação e processamento digital de sinais, estão inclusas no módulo, que é apresentado como uma caixa fechada, sendo aberta apenas como forma ilustrativa para a visão das funções e suas subfunções, pois está fora do escopo deste estudo os seus detalhamentos.

A Figura 40, apresenta o detalhamento do bloco radiotransmissor Tx, específico para a função de transmissão, extraído do *Simulink*[®] e sua respectiva referência em relação ao componente adotado neste estudo.

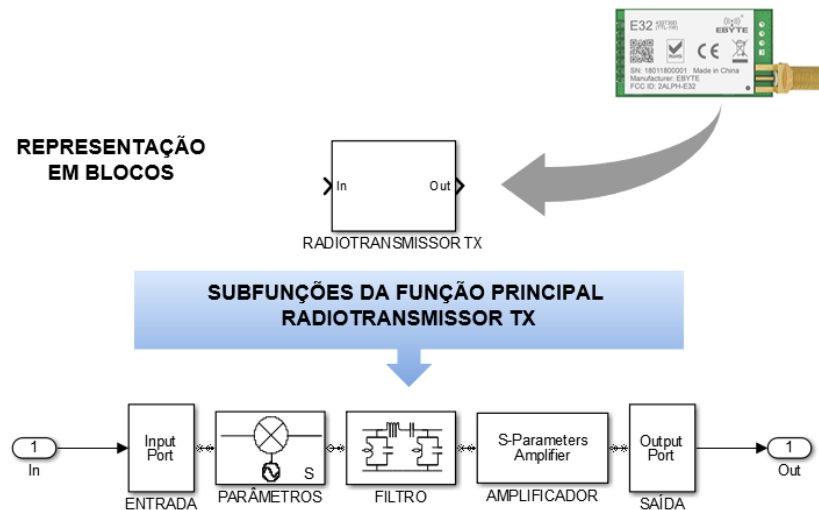


Figura 40 – Detalhamento do bloco radiotransmissor Tx.
Fonte Própria.

5.1.1.3 Bloco antena Tx e Rx

Os blocos das antenas, TX e RX, são tratados na mesma seção, visto que possuem comportamentos similares no cenário de simulação, com diferença específica relacionada ao valor do ganho de cada antena em relação a sua estação específica. A Figura 41 apresenta o bloco citado.

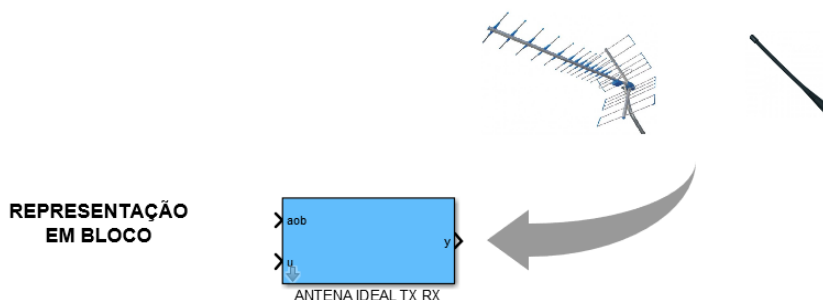


Figura 41 – Detalhamento do bloco antena Tx e Rx.
Fonte Própria.

Este bloco não possui desdobramentos em subfunções, seu comportamento é considerado como o de uma antena isotrópica ideal, com irradiação do sinal em 360°. Esta restrição se deve a limitações de disponibilidade, na biblioteca do *Simulink*[®], para a versão 2016a[®] adotada. Neste aspecto as condições de direcionalidade do lado da ET, não foram supridas na modelagem deste bloco, acarretando em impactos durante o processo de simulação, principalmente na questão do alcance de transmissão do sinal, conforme análise realizada na próxima seção. Entretanto, este bloco representa os componentes adotados no domínio específico, que considerou os modelos ATU-6B da fabricante Motorola, Tabela 56 e GY450 da fabricante GOBER, Tabela 57.

5.1.1.4 Bloco espaço livre

A modelagem do espaço livre é realizada em um bloco, cujo objetivo consiste em simular o canal de comunicação via satélite e as perdas provenientes de fatores externos. Conforme destacado na Seção 2.3, a propagação em espaço livre depende do caminho de visada direta entre o transmissor e o receptor, além de uma área desprovida de obstáculos ao longo do caminho. As comunicações via satélite experimentam este tipo de visada ideal, porém, situações como chuvas, nuvens, além de fatores como interferências de outras ondas de RF concorrentes, podem atrapalhar e proporcionar períodos de instabilidade (falta de comunicação) ou adição de ruídos ao canal de comunicação. O bloco da Figura 42, representa estas condições.

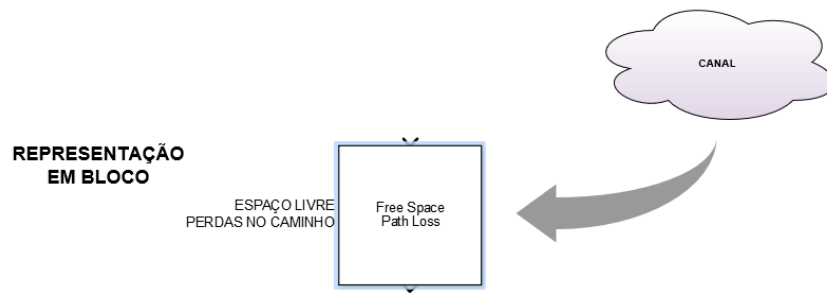


Figura 42 – Detalhamento do bloco espaço livre.

Fonte Própria.

Este bloco considera apenas as variáveis de frequência e distância para cálculo da perda no caminho, desconsiderando, em termos de simulação, os ganhos das antenas Tx e Rx. Esta restrição se deve a limitações de disponibilidade, na biblioteca do *Simulink*[®], para a versão 2016a adotada neste estudo. Uma comparação destes cálculos faz-se necessário para análise deste comportamento por meio das Equações (2.6) e ??Perda no espaço livre desconsiderando os ganhos). Considerando os valores presentes na Tabela 60, para as variáveis Frequência, Ganho Tx/Rx e Distância, pode-se chegar aos resultados para a perda no espaço livre ou atenuação no caminho.

Aplicando a Equação 2.6, que considera os ganhos das antenas de transmissão e recepção, obtêm-se o seguinte resultado:

$$PL(dB) = -10 \log \left[\frac{G_t G_r \lambda^2}{(4\pi)^2 d^2} \right];$$

$$PL(dB) = -10 \log \left[\frac{(10)(3)(0,7)^2}{(4\pi)^2 (10000)^2} \right];$$

$$PL(dB) = 90,3(dB).$$

Aplicando a Equação 2.7, que desconsidera os ganhos das antenas de transmissão e recepção, obtêm-se o seguinte resultado:

$$PL(dB) = 32,5 + 20 \log d + 20 \log f;$$

$$PL(dB) = 32,5 + 20 \log 10 + 20 \log 433e6;$$

$$PL(dB) = 225(dB).$$

A dispersão da energia pelo espaço é simbolizada pela perda no espaço livre ou atenuação do sinal, uma vez que a potência do sinal, que atinge o receptor, é menor que a potência transmitida. Analisando os resultados calculados para cada uma das condições apresentadas, nota-se que o modelo considerado no bloco de simulação, conforme mencionado, desconsidera os ganhos das antenas, e possui uma diferença de 134,7 dB que, em termos positivos, simboliza uma amplificação do sinal, sendo esta uma condição ideal presente na modelagem devido as restrições mencionadas anteriormente. Já em análise da condição real, que inclui os ganhos das antenas, a diferença em termos negativos simboliza a atenuação do sinal transmitido impactado pela relação distância *versus* frequência, fornecendo um resultado mais próximo da realidade aplicada.

Esta atenuação sofre outros impactos durante a transmissão, oriundos de interferências provenientes das alterações nas características do sinal, ocasionada por outros eventos exteriores ao sistema de transmissão, ou ainda, gerados por ruídos, produzidos por condições de natureza aleatória, em muitos casos pelos próprios equipamentos ativos utilizados nos sistemas de transmissão, neste caso de natureza térmica. Estas condições, além de seus valores, são de difícil previsão em um instante de tempo futuro, não sendo consideradas no cenário proposto deste estudo.

5.1.1.5 Bloco alvo

O Alvo simula um objeto móvel, no caso a EE, que é perpendicular à direção de deslocamento dos pulsos da ET, em termos do sinal incidente. O contrário também é válido, já que a EE se encontra em constante movimento durante o período da missão. Este bloco é apresentado na Figura 43.

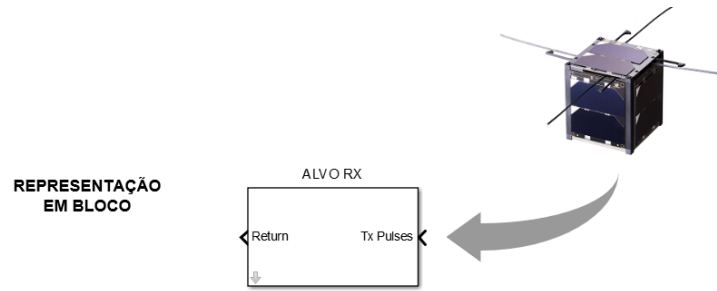


Figura 43 – Detalhamento do bloco alvo.
Fonte Própria.

As características desse bloco, analisa os parâmetros de distância, frequência, além de uma nova variável referente a velocidade do alvo, estipulada em 5 m/s, correspondente ao deslocamento durante a etapa de subida e descida da plataforma, considerando o transporte realizado por um balão estratosférico e, desconsiderando a variação da massa embarcada, valor ideal estabelecido pela equipe do projeto LAICAnSat.

5.1.1.6 Bloco radiotransmissor Rx

O componente Radiotransmissor E32-433T30D adotado neste estudo, possui aspectos de transmissão e recepção, por se tratar de um *transceiver*. Nestas condições, o bloco responsável pela recepção é representado separadamente, devido às características específicas para esta funcionalidade, com subfunções internas que abordam tratamentos específicos, referentes a: demodulação, demultiplexação, filtragem, amplificação, processamento digital de sinais e tratamento de ruídos, inclusos no módulo, e apresentado como uma caixa fechada, sendo aberta apenas como forma ilustrativa para a visão das funções e suas subfunções, pois os seus detalhamentos estão fora do escopo deste estudo.

A Figura 44, apresenta o detalhamento do bloco radiotransmissor Rx, específico para a função de recepção, extraído do *Simulink*[®].

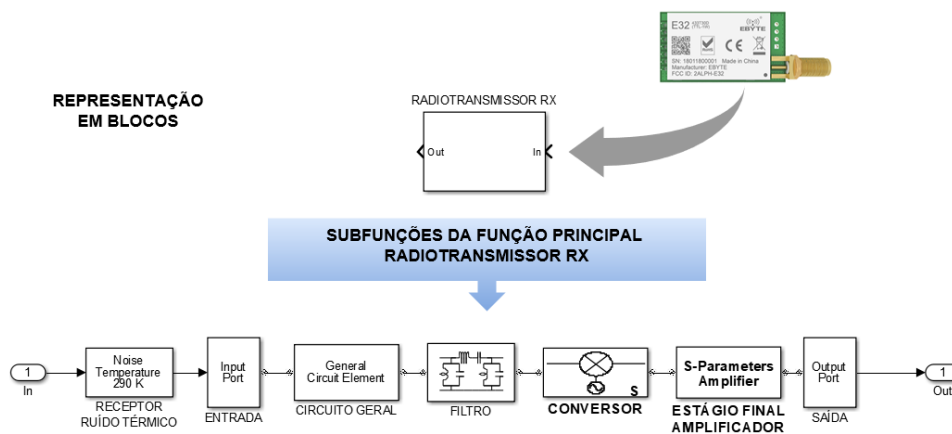


Figura 44 – Detalhamento do bloco radiotransmissor Rx.
Fonte Própria.

5.1.1.7 Modelo proposto

A partir dos blocos apresentados, foram realizadas as devidas ligações e integrações dos blocos em sequência, obtendo o modelo virtual simplificado para este estudo, apresentado na Figura 45.

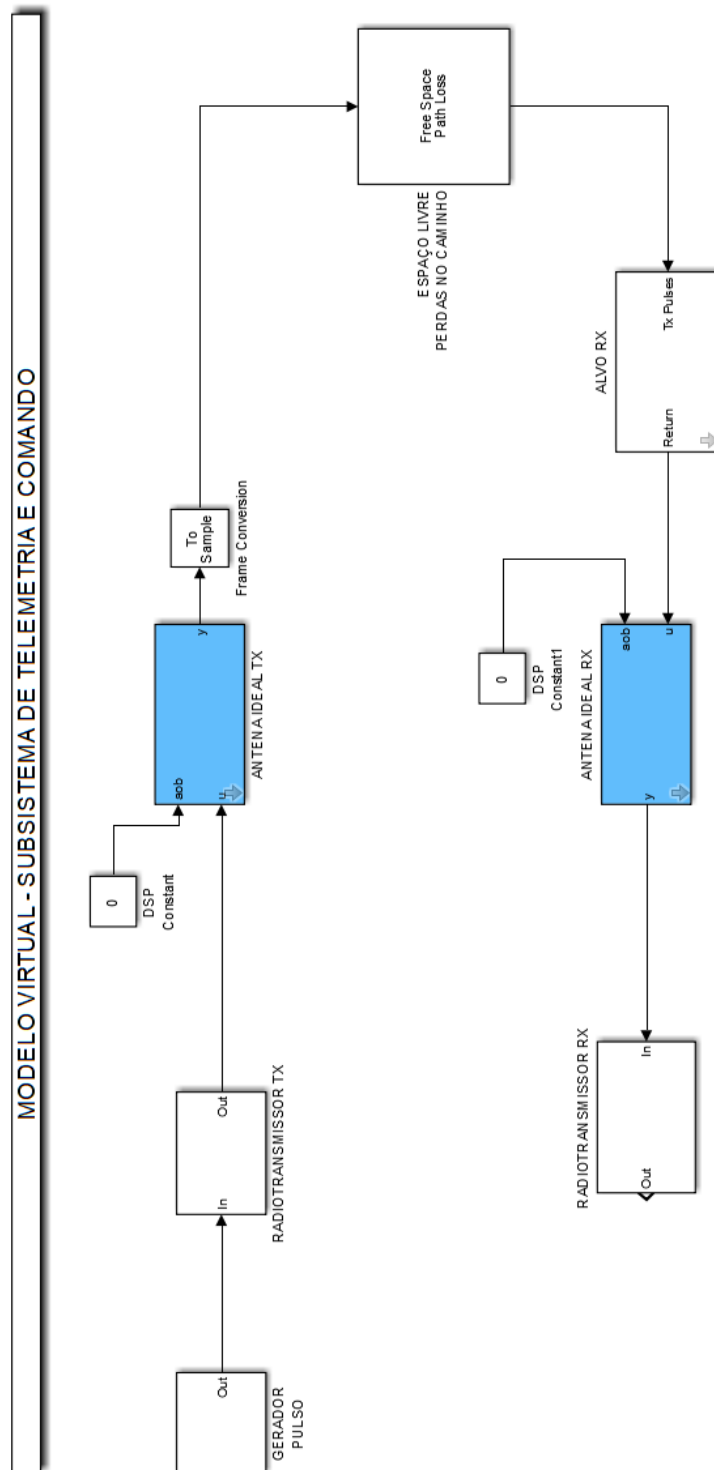


Figura 45 – Modelo Virtual simplificado.
Fonte Própria.

A Figura 45, ilustra uma simplificação do modelo, correspondente a um diagrama compostos pelos blocos principais, para a devida análise durante o processo de simulação, via compilação dos blocos e geração dos códigos importantes para coleta dos resultados, sendo fundamental a inclusão ao modelo de componentes que representem instrumentos ou periféricos de medidas, como por exemplo: analisadores de espectro, osciloscópios, pontas de prova entre outros.

As funcionalidades apresentadas no processo de modelagem, correspondentes aos componentes eletrônicos destinados ao *log* e interface homem-máquina (visor e monitor), no âmbito do modelo virtual, são representadas pelos analisadores de espectro e osciloscópios, interfaces responsáveis pelas saídas, e que apresentam as condições de operação do subsistema para análise do modelo. Em situações reais, o protótipo físico recebe o refinamento em *software* das variáveis operantes, proporcionando a tradução dos resultados obtidos para uma entrega em uma linguagem compreensível ao usuário.

A inclusão destes instrumentos ao modelo apoiam, na realização dos testes e verificações das condições de operação, em atendimento a fase MBD-6 da Matriz de Adoção do MBD, extraído da Tabela 12, rege a seguinte condição: *"O código de resultado do sistema de controle e o hardware usado na aplicação, são testados levando em consideração o modelo do objeto e o modelo do ambiente no qual o sistema é usado. A plataforma de destino e seus algoritmos são validados como teste de simulação e viabilidade computacional em regime de tempo real. Esses testes são essenciais para garantir a qualidade da solução e sua integridade a falhas."* para o caso em questão, os testes são simulados através do modelo virtual.

A Figura 46 traz uma visão completa do modelo virtual, contemplando as saídas, devidamente calibradas e dentro das escalas que apoiarão na apresentação dos gráficos durante a simulação da operação do modelo adotado, o Subsistema de Telemetria e Comando (T&C).

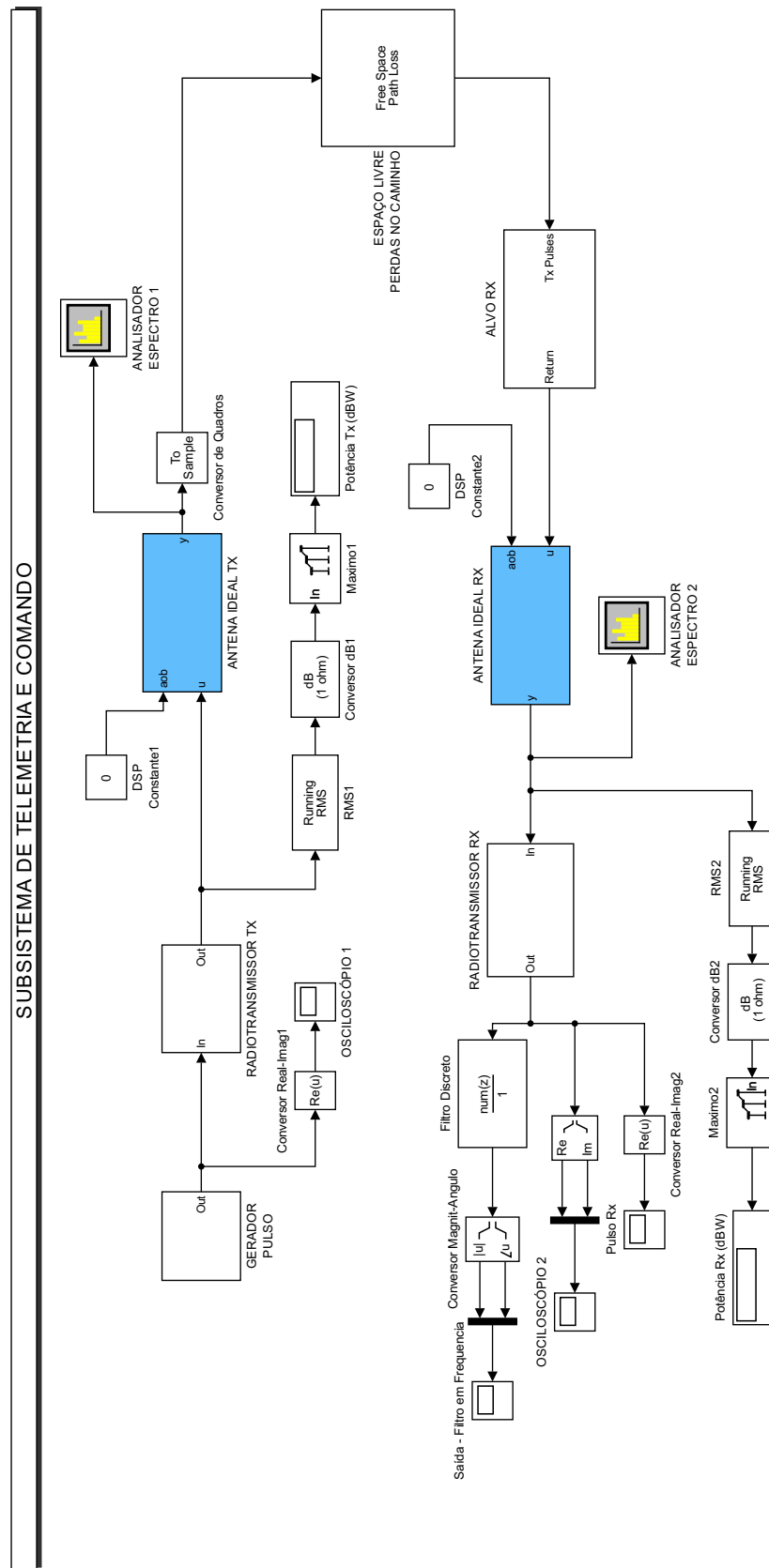


Figura 46 – Modelo Virtual completo.
Fonte Própria.

5.1.2 Teste MIL (*Model-In-The-Loop*)

O teste MIL tem a função de realizar simulações, por meio de iterações do modelo virtual elaborado. Estas repetições buscam analisar condições anormais provenientes das etapas de análise dos requisitos e elaboração da arquitetura referente aos modelos físicos e lógicos.

A compilação do modelo virtual apresentado na Figura 46, passou por várias iterações, onde as iniciais acusavam erros, impeditivos a continuidade do processo de compilação, os quais eram apontados pela ferramenta *Simulink*[®], em seguida corrigidos e novamente executados. Estes erros acusaram as seguintes condições:

1. erros de bibliotecas de blocos;
2. erros de integração dos blocos;
3. inconsistências de tipos de variáveis;
4. inconsistências de cálculos;
5. inconsistências de variáveis com respostas em *loop*;
6. anomalias nas saídas geradas.

Na proporção em que cada iteração era realizada, as condições listadas foram sendo tratadas progressivamente, melhorando os resultados do processo de simulação dinâmica, além de aproximar o modelo à resultados mais consistentes e dentro de limites aceitáveis. O modelo virtual proposto cumpriu as correções das condições listadas, até chegar no resultado considerado ideal para este estudo, assim como os requisitos e a arquitetura apresentadas no Capítulo 4, sendo consideradas apenas a versão final obtida para cada uma destas etapas.

A compreensão do comportamento das saídas dos blocos, foram analisadas com o auxílio de instrumentos de medidas representados por osciloscópios e analisadores de espectros, adicionados ao Modelo Virtual simulando as condições de operação do sistema, os quais contemplam as seguintes análises que acompanham o fluxo de operação desde a geração do pulso até a recepção.

- Análise do Gerador de Pulso;
- Análise do Espectro de Frequência Tx e Rx;
- Análise do Sinal Rx;

As próximas seções correspondem a estas análises, buscando esclarecer a simulação virtual de operação do sistema. Considerando para isto, um tempo estipulado de 120 segundos por simulação.

5.1.2.1 Análise do gerador de pulsos

O Gerador de Pulsos é responsável pela simulação de um sinal de frequência, com um ciclo de trabalho. Este componente corresponde as funções de telemetria ou comando, cumprindo a transmissão de um evento oriundo de um teclado ou em resposta a uma ação específica. O gráfico correspondente é apresentado na Figura 47.

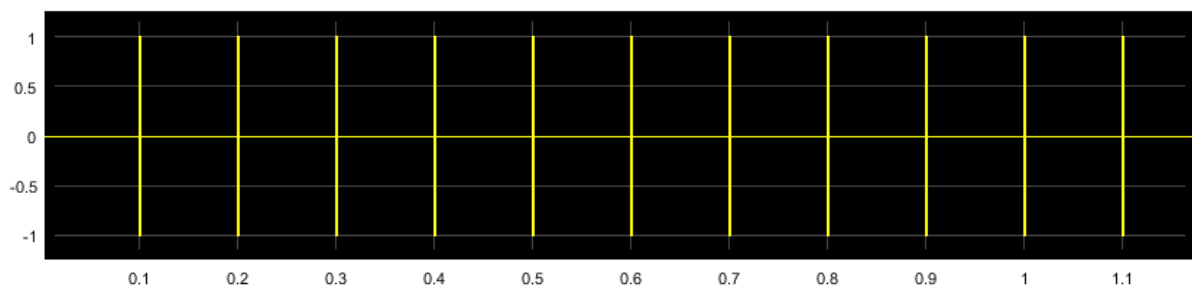


Figura 47 – Gráfico do gerador de pulso.
Fonte Própria.

O pulso é gerado em tempos periódicos determinados na parametrização do bloco, composto pela amplitude, além de variáveis de período e largura de pulso, modeladas na Tabela 60, conforme amostra extraída do tempo total.

O caminho realizado por este sinal passa pelo bloco do Radiotransmissor Tx e segue em direção a Antena Tx, propagando em um canal não-guiado, correspondente ao espaço livre, até atingir o alvo de destino e ser tratado pelo bloco do Radiotransmissor Rx.

5.1.2.2 Análise do espectro de frequência Tx e Rx

O Analisador de espectro é um instrumento eletrônico utilizado para verificação do comportamento das componentes harmônicas de sinais elétricos. No modelo virtual foram empregados dois analisadores, sendo um para o sinal transmitido e outro para o sinal recebido, visando esclarecer as condições das frequências e amplitudes, espalhadas no espectro de frequência de operação, estipulado na Tabela 60.

O resultado apresentado por estes instrumentos se encontra distribuído lado a lado na Figura 48, afim de facilitar o entendimento da transmissão, onde a composição do lado esquerdo está relacionada ao sinal transmitido e a resultante deste sinal, se encontra localizada do lado direito representando a recepção.

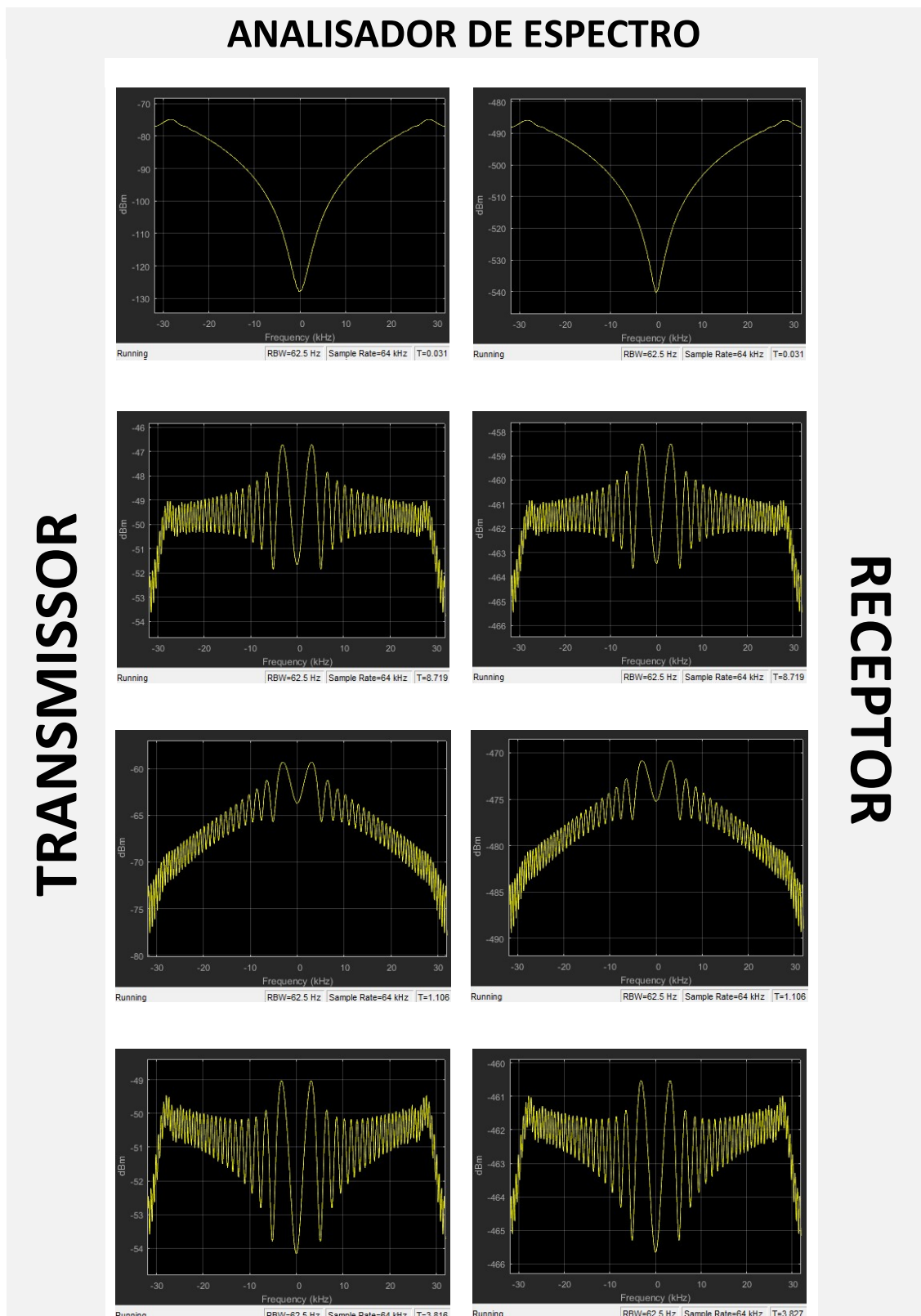


Figura 48 – Gráfico do analisador de frequência.
Fonte Própria.

Destaca-se, na Figura 48, que o sinal é variável para os diferentes pulsos gerados não sofrendo mudanças consideráveis no seu aspecto, comparando transmissão e recepção, mas em ocorrência da perda proporcionada pelo espaço livre devido a distância entre as estações, ocorre uma atenuação considerável do sinal na recepção, representado pelo eixo y, e isto tende a afetar a medida que a distância aumenta. Esta condição será melhor visualizada na próxima seção, onde serão consideradas diferentes distâncias de transmissão e o comportamento do sinal na recepção.

5.1.2.3 Análise do sinal Rx

Nesta etapa foram consideradas diferentes distâncias entre as estações, sendo: 4Km, 6Km e 10Km, o que permitiu uma análise comparativa do sinal recebido e respectivos comportamentos.

Antes de apresentar os sinais gerados pelo modelo virtual, é importante ressaltar que o modelo virtual adotado, não considera os ganhos das antenas Tx e Rx, devido a restrições comentadas anteriormente. Apesar desta condição existir no protótipo físico foram realizados os cálculos da potência recebida em relação a distância ($P_r(d)$), baseado na Equação de Friis (2.1), contemplando as duas situações.

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L}$$

Os ganhos adotados foram extraídos da Tabela 60, correspondentes aos modelos aplicados no domínio específico. Na consideração sem os ganhos, estas variáveis foram desconsideradas na equação. Estes cálculos estão disponíveis na Tabela 61.

Tabela 61 – Potência de recepção.

POTÊNCIA DE RECEPÇÃO			
DISTÂNCIA (KM)	SEM GANHO	COM GANHO	UNIDADE
4	0,20	5,80	mW
6	0,08	2,50	mW
10	0,03	0,93	mW

Fonte Própria.

O comparativo apresentado destes dados apoiou-se na compreensão dos gráficos e no entendimento da aplicação dos ganhos das antenas no contexto da transmissão. Observa-se, na Tabela 61, uma redução considerável na potência recebida ($P_r(d)$), quando não se considera o ganho das antenas, com tendência inversamente proporcional ao aumento da distância aplicada.

A Figura 49, apresenta o sinal completo em ordem crescente para as distâncias adotadas, medido no instrumento representado pelo bloco Osciloscópio 2, extraído do bloco Radiotransmissor Rx. Observa-se, nesta saída, duas componentes, nas cores azul e amarelo, referentes aos sinais de ruído e pulso respectivamente.

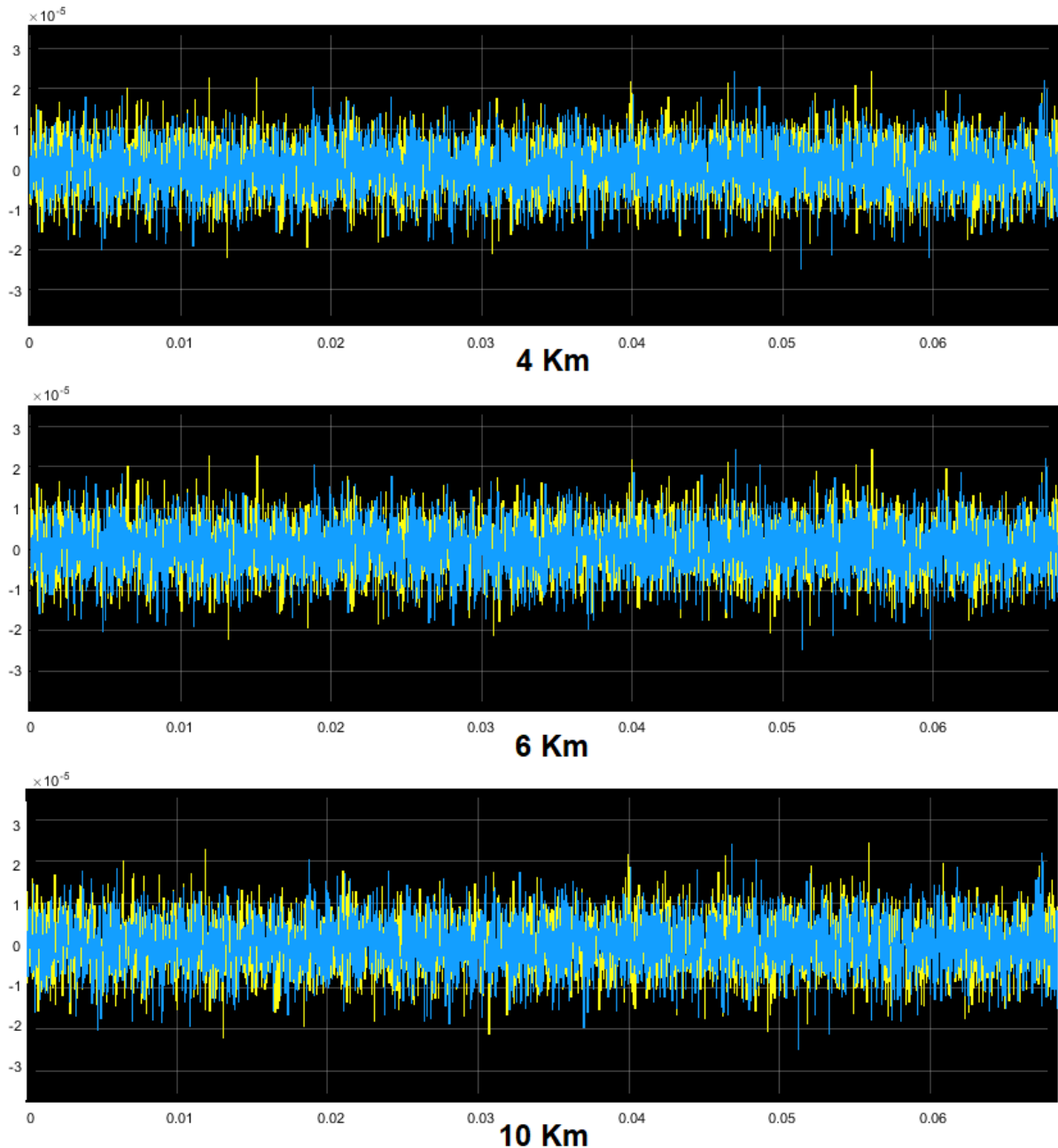


Figura 49 – Sinal completo na recepção.
Fonte Própria.

O sinal de interesse neste estudo corresponde a cor amarelo, referente ao pulso gerado. Dessa forma, a análise em conjunto dos sinais da Figura 49, dificulta a visualização e esclarecimento devido a sobreposição, sendo importante a separação dos sinais. A Figura 50,

inicia a apresentação deste processo, separando inicialmente o sinal de ruído térmico, na cor azul, através do bloco Saída Filtro, responsável por exibir a resposta do sinal ruidoso.

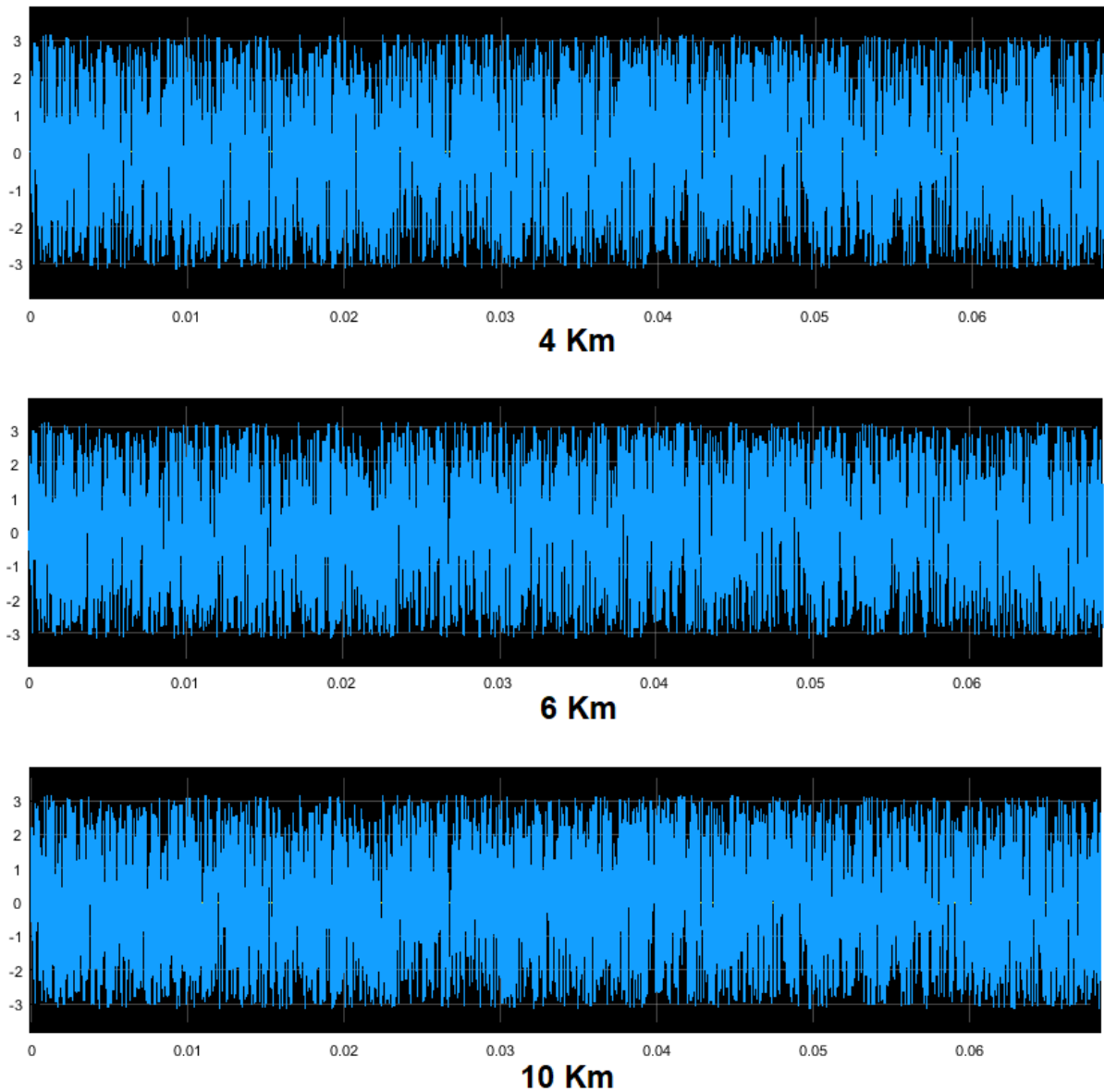


Figura 50 – Sinal proveniente de ruídos na transmissão.
Fonte Própria.

Os sinais ruidosos resultantes da Figura 50, originam-se dos equipamentos ativos utilizados nos sistemas de transmissão, além de interferências provenientes de fatores externos, devido a operação em frequência de 433 MHz, dentro da faixa destinada ao Serviço de Radioamador, o qual possui alto índice de utilização, acarretando em um espectro bastante poluído. Em uma análise detalhada do sinal, observa-se que à medida em que a distância aumenta, a envoltória do sinal apresenta um comportamento mais denso, relacionado à crescente interferência, ocorrida ao longo do percurso de transmissão, sendo suficiente verificar a minimização progressiva nos espaços vazios do eixo x. Conforme

discutido anteriormente, estas condições não serão abordadas com maiores detalhes neste estudo, mas esta visão geral é importante para a compreensão desta variável, que oferece perda no desempenho durante a operação.

Finalmente, a Figura 51, apresenta o sinal de pulso filtrado sem a presença do ruído, para a devida análise.

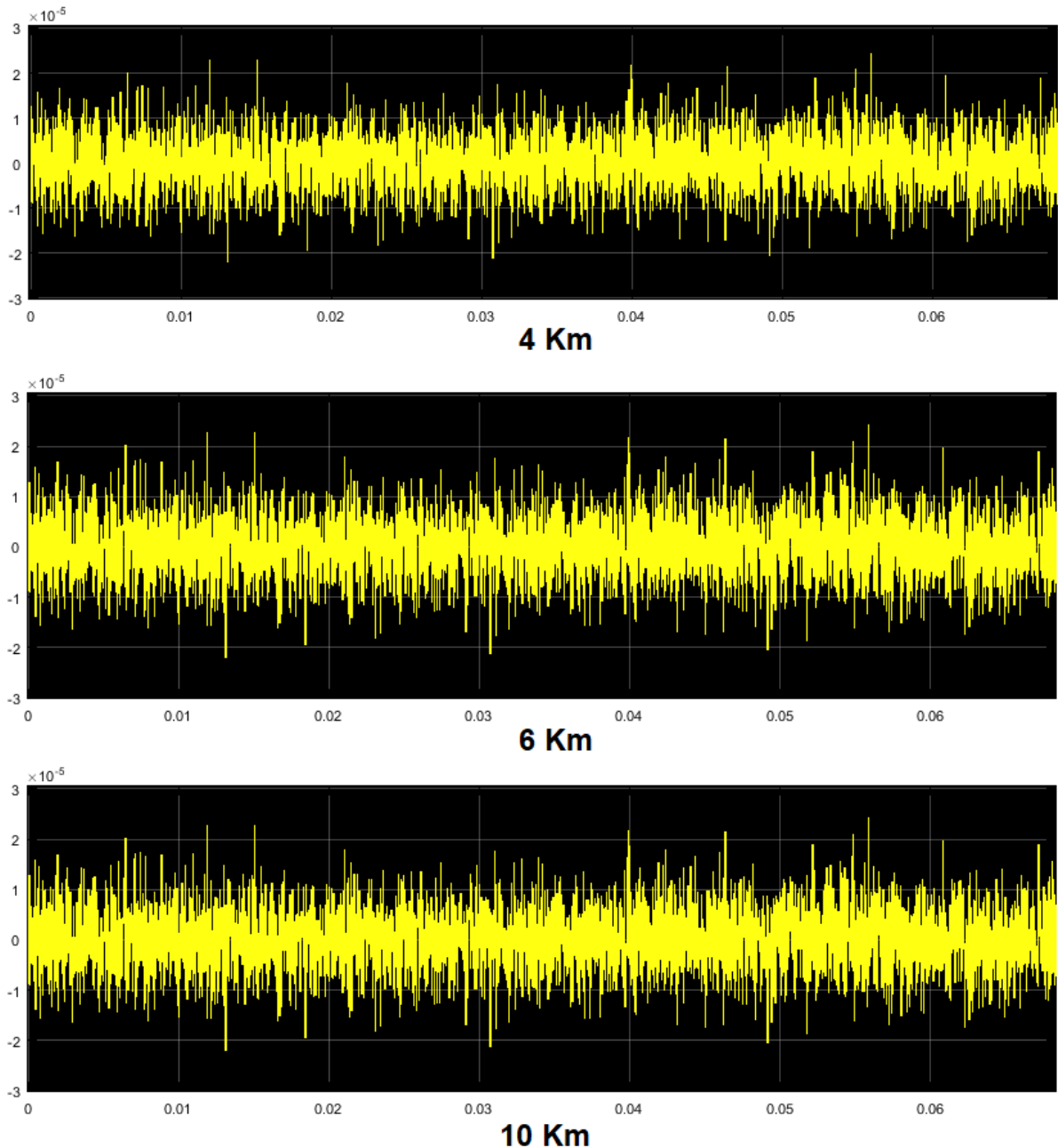


Figura 51 – Sinal de interesse na recepção
Fonte Própria.

Os sinais recebidos, para as três distâncias empregadas, possuem grande similaridade, o que comprova a entrega do pulso na recepção. Outras análises são importantes, a primeira foi levantada durante a análise da Figura 50, vinculada ao tratamento do

sinal realizado no bloco Radiotransmissor Rx, com a aplicação de métodos de filtragem, decodificação e regeneração do sinal, lançando mão de processos de demodulação, decodificação, processamento digital de sinais, amplificação, dentre outros fatores que auxiliam na extração do sinal desejado em detrimento dos ruídos provenientes de interferências, desvanecimentos e perdas adicionadas ao longo do percurso de transmissão, atenuando a potência do sinal e o seu conseqüente alcance.

Pensando nestes impactos, são consideradas variáveis importantes na obtenção de uma transmissão estável, a potência de transmissão, a largura de banda, o balanceamento de impedância, diretividade, ganho das antenas de transmissão, qualidade do *hardware* e dos conectores e acoplamentos ligados ao sistema, relacionado à compatibilidade eletromagnética. Muitas das questões relacionadas às técnicas citadas não foram consideradas detalhadamente neste estudo, mas se faz necessário abordar de forma indireta, visto que são elementos inseridos na maioria dos componentes COTS pesquisados e, conseqüentemente, presentes nas subfunções dos blocos correspondentes ao domínio específico da eletrônica, simbolizado pelo Radiotransmissor Tx e Rx, conforme apresentado nas Figuras 40 e 44

5.1.3 Teste SIL (*Software-In-The-Loop*)

No teste SIL, o modelo virtual do *hardware*, apresentado em forma de blocos e discutido na Seção 5.1.1, foi transformado em código, gerado na Linguagem C. Esta simulação consiste no estágio onde a ferramenta de geração automática de código fornece as estratégias de operações estabelecidas no MIL automaticamente em código. Aqui o modelo aproxima-se mais do modo real de operação.

Neste teste, a compilação do modelo virtual e geração do código para o *hardware* envolvido, passou por iterações do modelo, apesar de um número menor de repetições em relação ao MIL, mas já é possível observar erros de codificação, impeditivos a continuidade do processo de geração do código, apontados pela ferramenta *Simulink*[®], e posteriormente corrigidos e novamente executados. Estes erros acusaram as seguintes condições:

1. erros de instâncias de variáveis;
2. erros de integridade de tipos de variáveis;
3. inconsistências em conexões dos blocos;
4. anomalias na execução do modelo.

Na proporção em que cada iteração era realizada, as condições listadas foram sendo tratadas progressivamente, aprimorando a eficiência de execução do código gerado. Durante esta etapa foi selecionado o modelo do *hardware* para a Interface Microcontrolada, a ser embarcado o código gerado. Essa escolha atendeu ao processo de modelagem do

domínio específico realizado na Seção 4.2.2.1. Dessa forma, o modelo ATMEGA328 (8-bit) da fabricante ATMEL supriu os parâmetros específicos de *hardware* selecionado neste processo.

O fluxograma desta fase pode ser observado na Figura 52. Conforme ilustrado, o processo é representado pelo código em linguagem C e o modelo segue sendo representado pelo diagrama de blocos, correspondente ao modelo virtual.

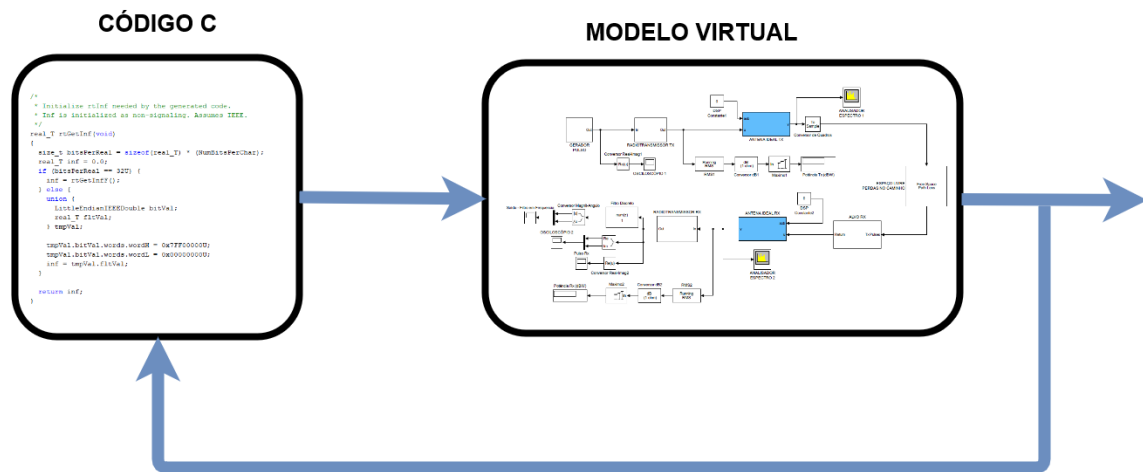


Figura 52 – Fluxograma teste SIL.
Fonte Própria.

Os *loops* gerados buscaram refinar o código C, para melhoria da abstração do processo, contemplando todas as funcionalidades de *hardware* exigidas, configurando as bibliotecas, as tomadas de decisão impostas e parametrização de variáveis e constantes do sistema.

O Apêndice B, detalha todo o código gerado, referente ao arquivo ModeloVirtual.c, após as condições e restrições impostas pelo sistema e ajustadas via *Simulink*[®], o código em Linguagem C foi gerado sem erros, compondo o *hardware* embarcado.

5.1.4 Teste HIL (*Hardware-In-The-Loop*)

Nesta etapa, o processo gerado em Linguagem C se encontra instalado no sistema final ou protótipo, interagindo apenas com a planta através das entradas apropriadas do processo. No caso específico deste estudo, a simulação do modelo é finalizada nesta etapa, transferindo os testes para o protótipo físico, em atendimento a modelagem proposta na Figura 35, onde os componentes dos domínios mecânicos e eletrônicos foram adquiridos e conectados, física e logicamente para composição do protótipo. Importante destacar que o subsistema de telemetria e comando, possui dois conjuntos de componentes, que constituem a estação embarcada (EE) e a estação terrestre (ET), cujas respectivas integrações dos domínios físicos e lógicos são apresentadas nas próximas seções.

5.1.4.1 Integração física - Estação Embarcada

O Subsistema de Telemetria e Comando da EE diz respeito ao conjunto embarcado na plataforma estratosférica composta pelos demais subsistemas, sendo assim, a sua integração e montagem ocorrem nas ocasiões de lançamento da plataforma. Esta condição é representada na Figura 53.

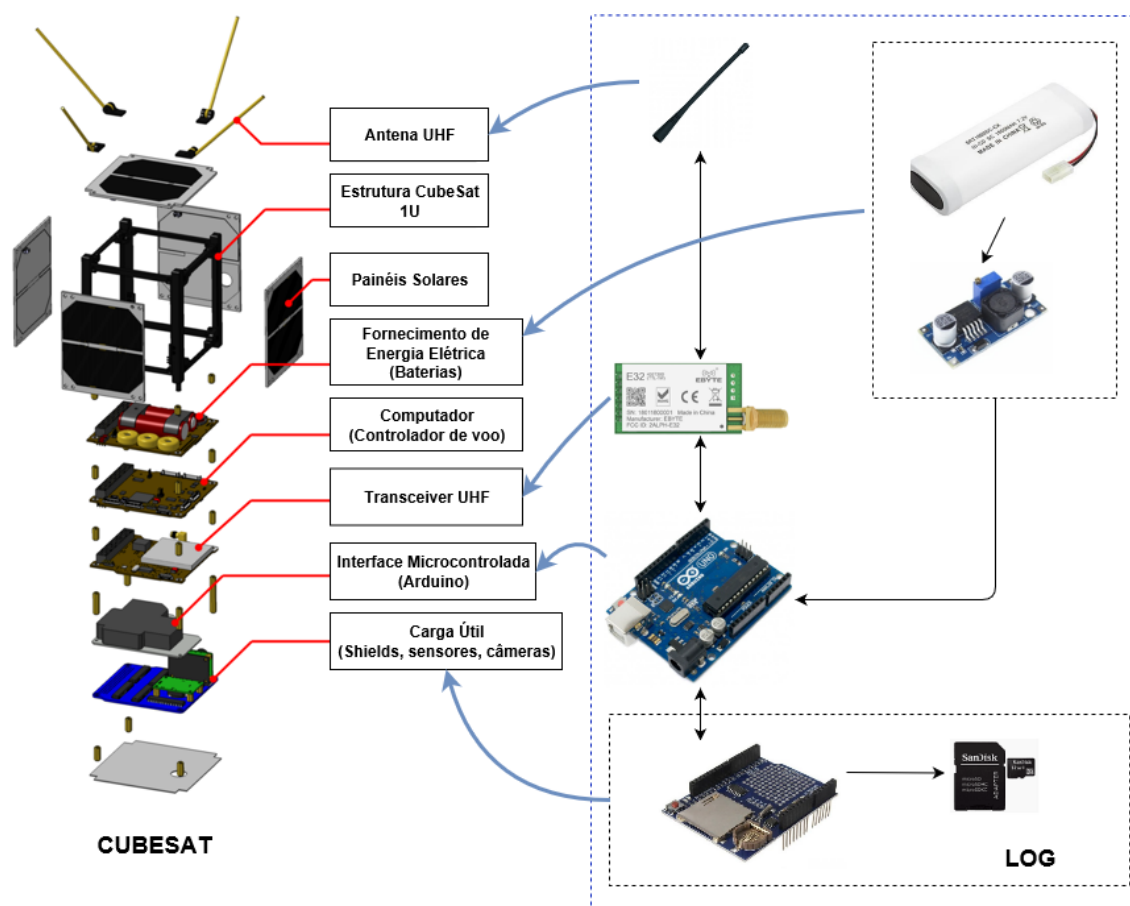


Figura 53 – Integração componentes EE - Plataforma.
Fonte Própria.

A Figura 53 fornece uma visão macro, contemplando um exemplo da distribuição e posicionamento dos componentes embarcados em uma plataforma estratosférica do tipo CubeSat 1U, com ênfase para os elementos que compõem o domínio físico, proposto na arquitetura deste estudo.

Faz-se necessário uma visão detalhada das conexões físicas para esclarecimento das ligações que integram os componentes, representado pelo diagrama elétrico da Figura 54.

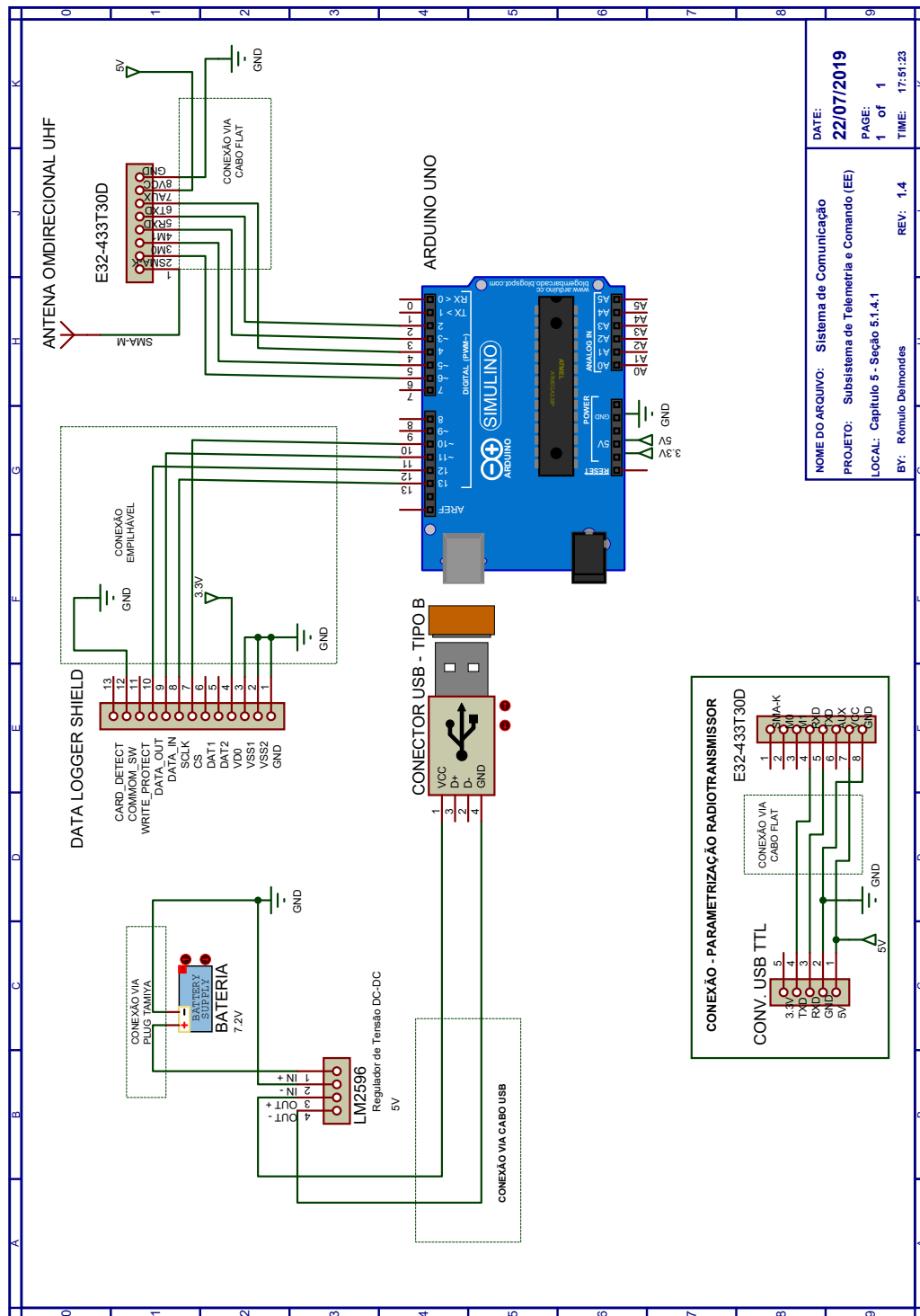


Figura 54 – Diagrama físico EE.
Fonte Própria.

A elaboração do diagrama elétrico, Figura 54, adotou como ferramenta de apoio o *Proteus Design Suite Versão 8*. Este *software* permitiu interligar os componentes físicos, através de cabos e conexões específicas, conforme apontamentos realizados nas conexões, com o objetivo de atender as condições de operação dentro de padrões aceitáveis, minimizando a possibilidade de interferências ocasionada pela incompatibilidade eletromagnética.

Em continuidade à análise da Figura 54, observa-se na parte inferior do desenho, um quadro nomeado Parametrização Radiotransmissor, este diagrama apresenta as conexões entre o conector USB TTL - presente no domínio eletrônico Tabela 54 - e o radiotransmissor modelo E32-433T30D, adotado neste estudo. Esta integração permite a parametrização das variáveis de funcionamento do radiotransmissor, via interface externa (Notebook, Laptop, microcomputador, entre outros), através do *software RF Setting Versão 0.70*, fornecido pelo fabricante EByte. Tal configuração de parâmetros deve, obrigatoriamente, ser realizado para ambas estações, sempre precedendo a realização da missão. Estas serão detalhadas adiante, no processo de análise da integração lógica do sistema.

5.1.4.2 Integração lógica - Estação Embarcada

A integração lógica, corresponde a etapa do comportamento do sistema e seus periféricos, aborda questões que tangem a comunicação, operação e integridade dos componentes físicos, além de fatores condicionais que compreendem fluxos para tomada de decisão, desempenho, segurança, normatização e usabilidade do sistema, quesitos apontados durante as etapas de levantamento dos requisitos não-funcionais e compreendidos na modelagem da arquitetura lógica, incorporados através das respectivas funções de processamento e controle das estações, conforme apresentado no Diagrama de Fluxo da Arquitetura (DFA) (Figura 34).

No âmbito da EE, pode-se extrair do DFA apenas a parte específica para a integração lógica desta estação, a qual envolve as funções para interface, processamento, controle, manutenção e diagnóstico, conforme apresenta a Figura 55. Cada um dos subsistemas ilustrados, representam as funções levantadas nas fases de requisitos, e modeladas durante a elaboração do modelo virtual no cenário que envolve o *hardware*, a transmissão e recepção dos dados, viabilizado pelos testes MIL e SIL, e possibilitado pelas iterações da análise do comportamento do sistema para o primeiro teste e pela geração de código de máquina para o segundo teste. Através do código fornecido, foi possível realizar o refinamento e consequente adaptação à realidade pretendida.

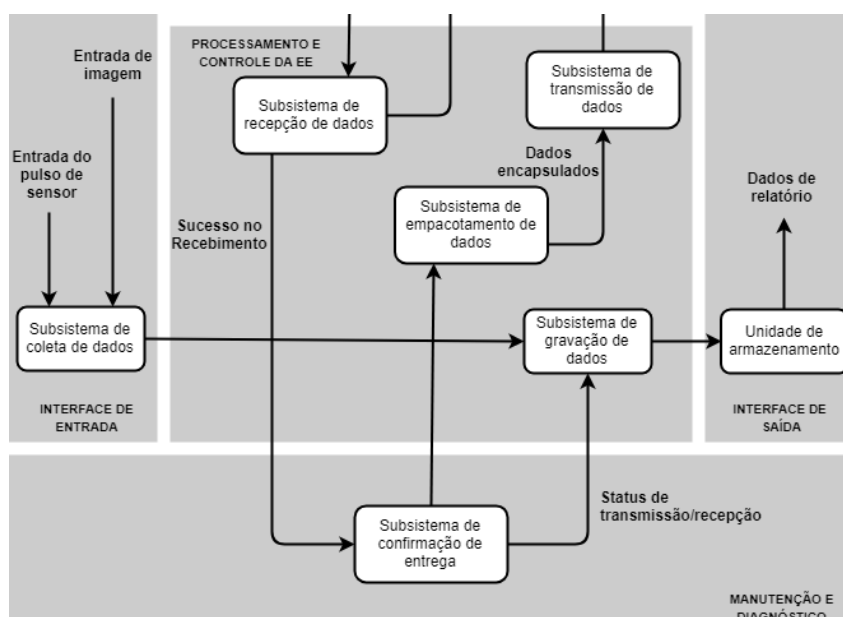


Figura 55 – Diagrama de integração lógica EE.
Fonte Própria.

Quanto aos demais subsistemas, não foram englobados no modelo, como por exemplo: Empacotamento de Dados, Gravação de Dados e Confirmação de Entrega, foram desenvolvidos à parte e integrados ao código principal. O subsistema de Coleta de Dados, não chegou a ser desenvolvido neste estudo, mas a sua integração pode ser observada através do subsistema Confirmação de Entrega, visto que relacionam, no âmbito da resposta, a um comando oriundo da ET, por meio do envio de mensagens ou dados provenientes de uma ação de resposta oriunda da interface microcontrolada embarcada na EE.

As funções correspondentes aos subsistemas presentes na arquitetura lógica da EE, compõem a nomenclatura apresentada na página 108 e foram refinadas para integração à interface microcontrolada. Este processo de refinamento do código, utilizou como ferramenta de apoio o ambiente de desenvolvimento *Arduino IDE Versão 1.8.9*, embarcada, em tempo real, ao componente físico representado pelo modelo Arduino Uno (Interface Microcontrolada), composta pela microprocessador da fabricante ATMEL, modelo AT-MEGA328 (8-bit), cujo cenário de compilação e simulação ocorre diretamente no *hardware*, definido no modelo virtual em atendimento às conexões da integração física.

O código fonte desenvolvido, com suas respectivas funções específicas, bibliotecas e condições lógicas, que compõem a estação embarcada, são apresentados no Apêndice C para o entendimento dos resultados obtidos durante os testes práticos, realizado após a integração completa do sistema de comunicação.

As seções a seguir, em similaridade à integração proposta para a Estação Embarcada, realizará os mesmos detalhamentos com suas respectivas particularidades para a Estação Terrestre.

5.1.4.3 Integração física - Estação Terrestre

O Subsistema de Telemetria e Comando da ET, diz respeito ao conjunto de componentes que compõem a unidade terrestre da plataforma estratosférica. Esta composição é similar a uma central de comando e monitoramento, exigindo condições de operação e IHM que proporcionem um certo nível de interação durante a ocasião de lançamento da plataforma. Em virtude destes detalhes, é de grande importância que ocorra um certo nível de consonância entre os componentes envolvidos, os quais são apresentados na Figura 56, extraída da arquitetura do domínio físico elaborada no Capítulo 4, com resultado proposto na Tabela 59.

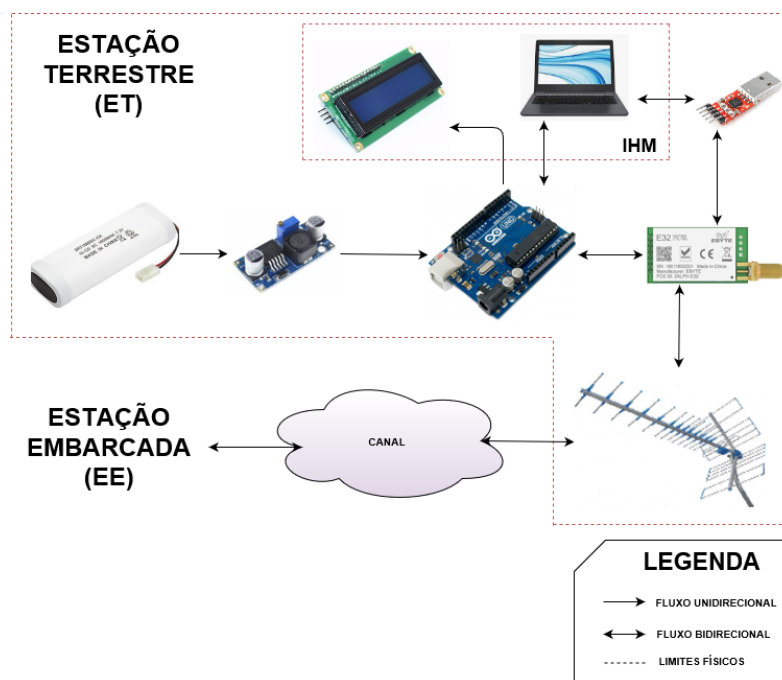


Figura 56 – Integração componentes ET - Plataforma.
Fonte Própria.

Esta consonância diz respeito à integração dos equipamentos físicos, em análise a Figura 53, que apresentou os componentes envolvidos na EE. Nota-se que a Estação Embarcada possui uma estrutura para acomodação e embarque dos componentes específicos, os quais são devidamente integrados à unidade CubeSat. Quanto à ET, faz-se necessário pensar em uma unidade de condicionamento funcional dos componentes, que permita fornecer condições para atendimento dos requisitos modelados para o sistema, como: dimensões (Tabela 19), peso (Tabela 20), controlabilidade (Tabela 21), precisão (Tabela 22), flexibilização (Tabela 23), exatidão (Tabela 29), monitoração (Tabela 34), compatibilidade de conexão (Tabela 35), versatilidade (Tabela 36). Considerando este cenário, é importante detalhar as conexões físicas para esclarecimento das ligações que integram os componentes da ET, representado pelo diagrama elétrico da Figura 57.

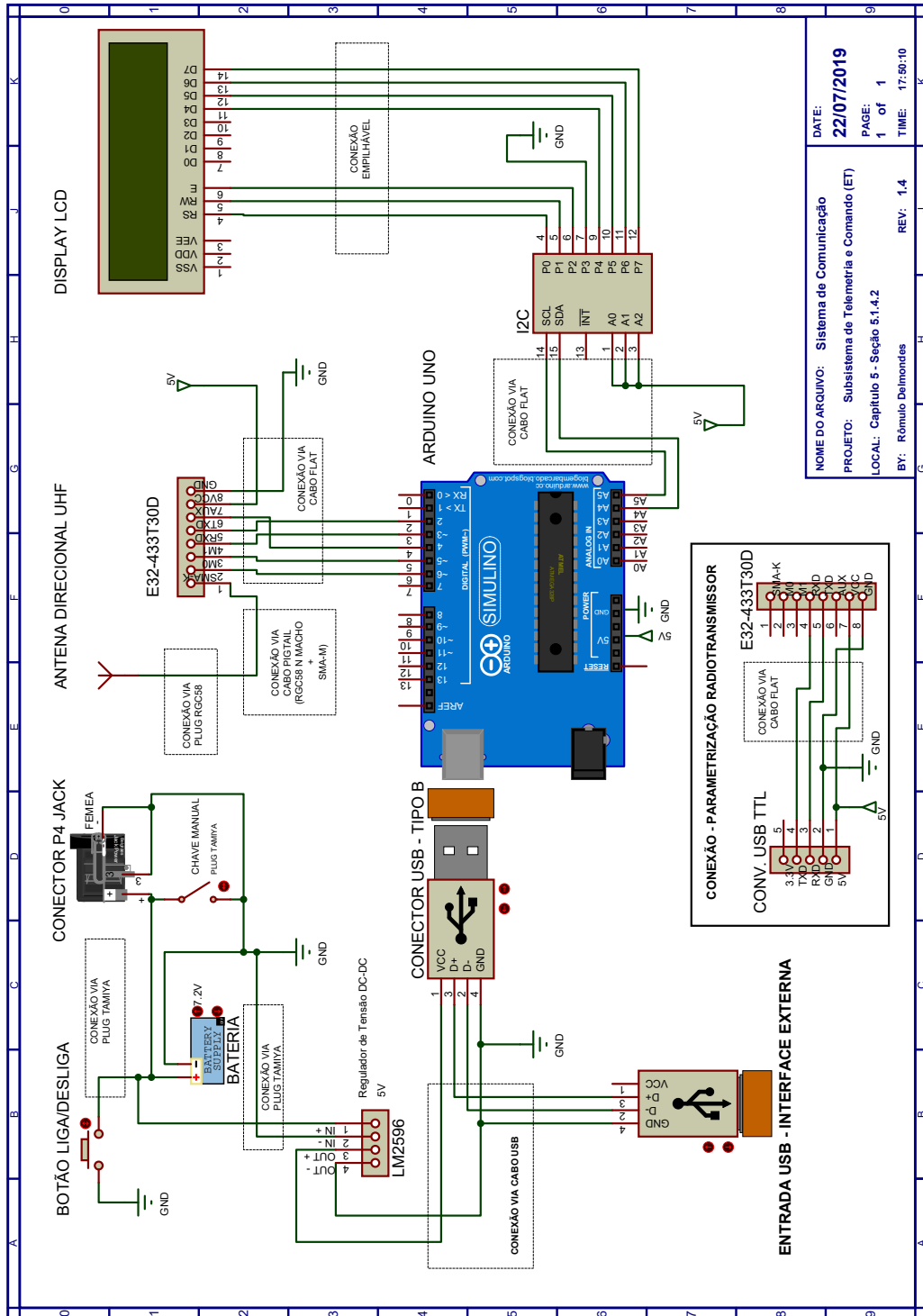


Figura 57 – Diagrama físico ET.
Fonte Própria.

A elaboração do diagrama elétrico da Figura 57, permitiu interligar os componentes físicos através de cabos e conexões específicos, conforme apontamentos realizados nos retângulos pontilhados que representam os tipos de conexões adotados, com o objetivo de atender as condições de operação dentro de padrões aceitáveis, minimizando a possibilidade de interferências ocasionada pela incompatibilidade eletromagnética.

Observa-se também, na parte inferior da Figura 57, um quadro similar ao apresentado do Diagrama Físico da EE (Figura 54). Nomeado como Parametrização Radiotransmissor, este diagrama apresenta as conexões entre o conector USB TTL - presente no domínio eletrônico Tabela 54 - e o radiotransmissor modelo E32-433T30D, adotado neste estudo, cujo detalhamento dos motivos de sua presença foram expostos na seção correspondente, sobre a integração física da EE.

Com o apoio do diagrama físico, a próxima etapa foi compor a unidade para condicionamento dos equipamentos, visando o atendimento aos requisitos funcionais citados anteriormente, esta necessidade agrega fatores importantes de usabilidade, praticidade e segurança dos dispositivos eletrônicos e foi pensada no âmbito da integração e facilidade de uso, ou seja, afim de garantir condições de operações apropriadas para o conjunto da ET e permitir ao usuário um fácil manejo da tecnologia. Assim, os equipamentos da estação terrestre foram encapsulados em uma interface que integra saídas IHM (Display LCD e Entrada USB), chave seletora liga-desliga, além de entradas para conexão da antena transmissora e carregamento externo da bateria.

As ligações seguiram exatamente as condições impostas pelo diagrama elétrico da Figura 57, com as respectivas conexões seguindo os padrões adotados pelos fabricantes. Lembrando que todos os equipamentos, tratam-se de componentes de prateleira, seguindo a tendência COTS. Desta forma as interfaces de entradas e saídas atendem as especificações dos fabricantes, onde no âmbito do desenvolvimento desta interface integrada para a ET, não houve a preocupação em projetar conexões específicas, sendo apenas realizadas as ligações, em atendimento aos conectores adotados nos modelos escolhidos para compor o domínio físico.

A Figura 58 ilustra o desenho proposto da Unidade de Condicionamento Funcional da Estação Terrestre, denominada neste estudo de UCF-ET, interface remodelada a partir da caixa hermética, apresentada como solução na Seção 4.2.3.3, Tabela 58, a qual foi devidamente adaptada, com o auxílio do *software SolidWorks Versão 2018*, para atender às condições de integração do domínio eletrônica para a ET.

A Figura 59 trás a vista explodida que compõe a UCF-ET, compreendendo toda a integração física do domínio eletrônica, apresentando os componentes que compõem a unidade e suas respectivas interfaces de comunicação com o ambiente externo. Maiores informações a respeito desta unidade estão disponíveis no Apêndice E, com detalhes sobre sua integração como interface do usuário, conexão a antena e detalhes da nomenclatura.

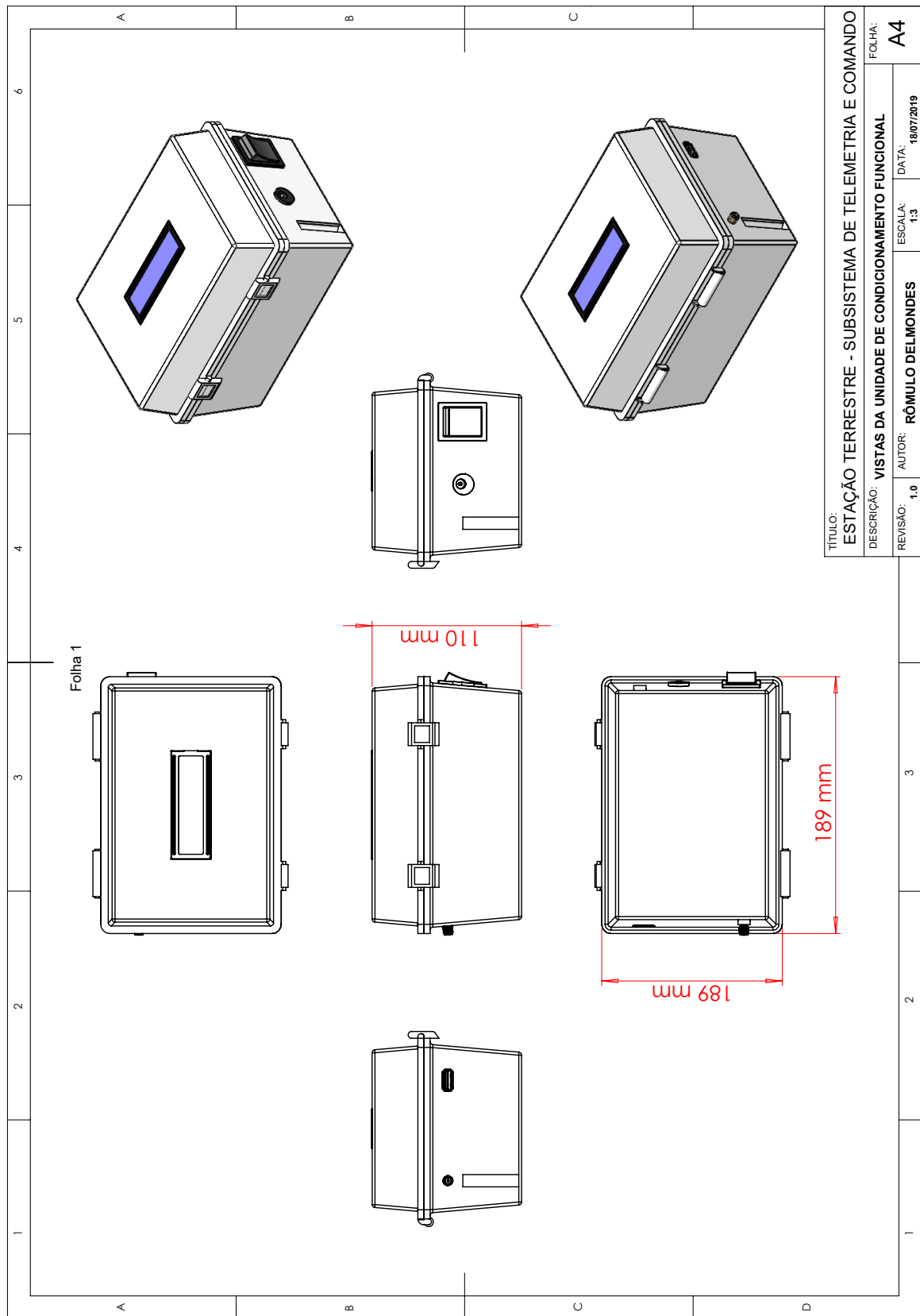


Figura 58 – Diagrama da Unidade de Condicionamento Funcional.
Fonte Própria.

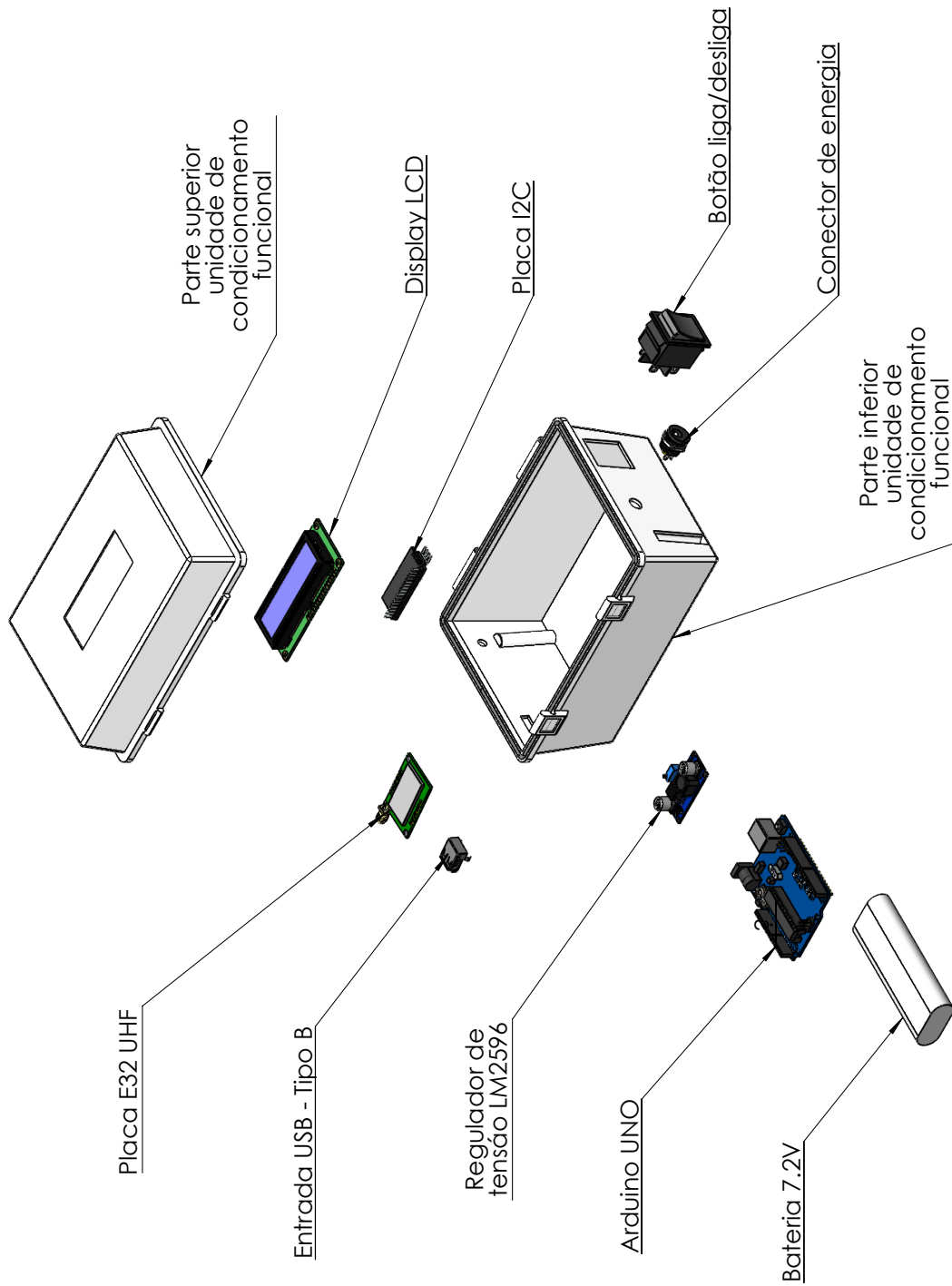


Figura 59 – UCF - Integração Domínio eletrônico.
Fonte Própria.

5.1.4.4 Integração lógica - Estação Terrestre

A integração lógica, pensada para o âmbito da ET, envolve as funções para interface, processamento, controle, manutenção e diagnóstico, conforme apresenta a Figura 60. Cada um dos subsistemas ilustrados, representam as funções levantadas nas fases de requisitos, modeladas durante a elaboração do modelo virtual no cenário envolvendo o *hardware*, a transmissão e recepção dos dados, viabilizado pelos testes MIL e SIL, possibilitado pelas iterações da análise do comportamento do sistema para o primeiro teste e pela geração de código de máquina para o segundo teste. Através do código fornecido, foi possível realizar o refinamento e consequente adaptação à realidade pretendida.

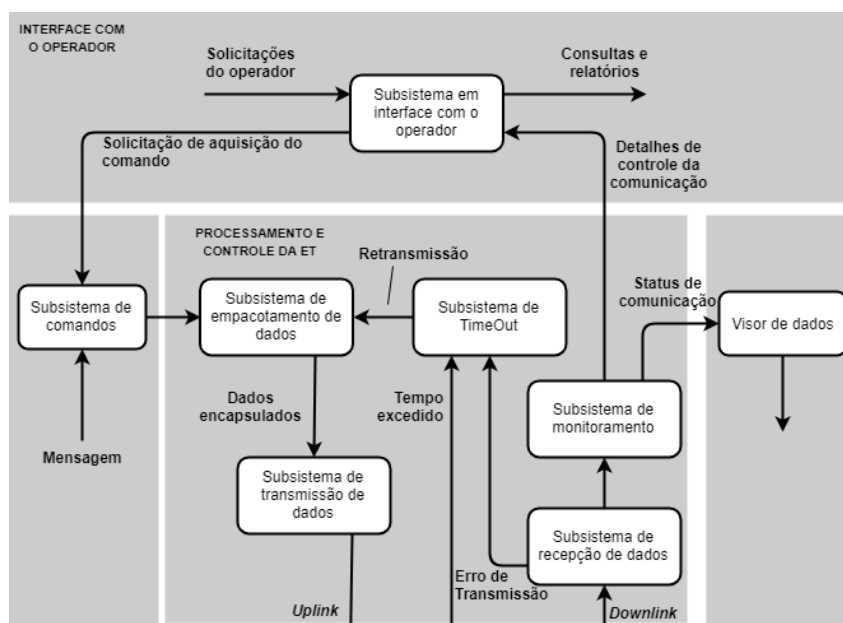


Figura 60 – Diagrama de integração lógica ET.

Fonte Própria.

Quanto aos demais subsistemas, que não foram englobados no modelo, como por exemplo: Empacotamento de Dados, Timeout e Monitoramento, foram desenvolvidos à parte e integrados ao código principal. O subsistema de Interface com o operador, somente a etapa de informação via texto foi implementada, sendo necessário uma futura adequação do código para tornar o ambiente visual intuitivo, possibilitando maior interatividade.

O Subsistema de Comando não chegou a ser desenvolvido neste estudo, mas a sua integração pode ser observada através do envio da mensagem, que esta implementada na estrutura do pacote, simulando o envio de um comando oriundo da interface microcontrolada da ET.

As funções correspondentes aos subsistemas, presentes na arquitetura lógica da ET, compõem a nomenclatura apresentada na página 108 e foram refinadas para integração à interface microcontrolada. Este processo de refinamento do código, é similar ao da integração lógica da EE, conforme descrito anteriormente. Utilizou-se como ferramenta de

apoio, o ambiente de desenvolvimento *Arduino IDE Versão 1.8.9*, embarcado, em tempo real, ao componente físico representado pelo modelo Arduino Uno (Interface Microcontrolada), composta pela microprocessador da fabricante ATMEL, modelo ATMEGA328 (8-bit), cujo cenário de compilação e simulação é feito direto no *hardware* definido no modelo virtual em atendimento as conexões da integração física.

O código fonte desenvolvido, com suas respectivas funções específicas, bibliotecas e condições lógicas, que compõem a estação terrestre, são apresentados no Apêndice D para o entendimento dos resultados obtidos durante os testes práticos, realizados após a integração completa do sistema de comunicação.

A próxima seção apresenta o protótipo resultante, obtido através da integração física dos componentes correspondentes a cada uma das estações.

5.1.4.5 Protótipo final

A análise das integrações físicas e lógicas, resultaram nos respectivos protótipos, correspondentes a cada uma das estações, os quais foram montados em bancada para composição da proposta referente a esta pesquisa.

A configuração resultante para a EE, é apresentada na Figura 61, esta ilustração equivale a montagem da estrutura física. As condições operacionais, referentes ao *software* embarcado, serão analisadas adiante no teste prático realizado.

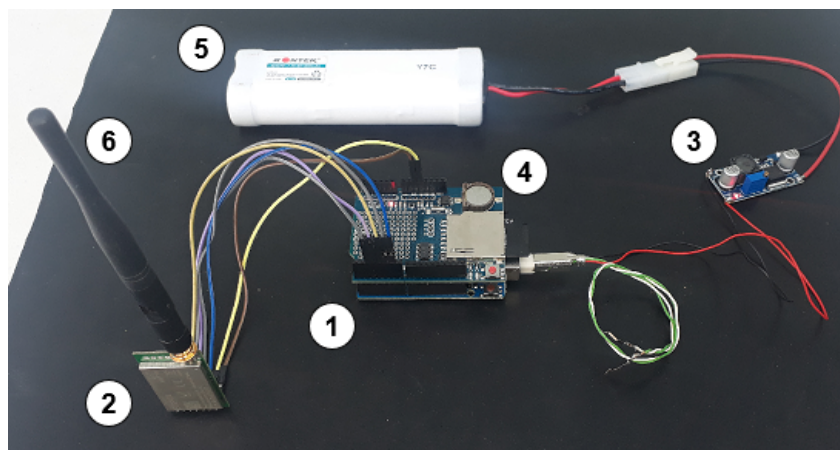


Figura 61 – Protótipo final EE.
Fonte Própria.

Observa-se, na Figura 61, a presença dos respectivos componentes, modelados durante a etapa de análise do domínio específico e presente na Tabela 59, etapa em que foram definidas as soluções escolhidas para composição do domínio físico, referente a parte mecânica e eletrônica. Os componentes foram listados, conforme numeração presente na ilustração, em atendimento a integração dos componentes embarcados e vinculados

a plataforma, conforme Figura 53 e em consonância as ligações e conexões elétricas apresentadas no Diagrama Físico da Figura 54.

1. Interface Microcontrolada - Modelo Arduino Uno;
2. Radiotransmissor - Modelo E32-433T30D;
3. Componente Eletrônico - Modelo LM2596;
4. Componente Eletrônico - Modelo Data Logger Shield;
5. Bateria - Modelo SC;
6. Antena Omnidirecional - Modelo ATU-6B.

No cenário da ET, a configuração resultante é apresentada na Figura 62. A composição interna da unidade, para condicionamento dos equipamentos, foram montados de maneira a atender a composição das ligações e conexões físicas. As condições operacionais, referentes ao *software* embarcado, serão analisadas adiante no teste prático realizado.

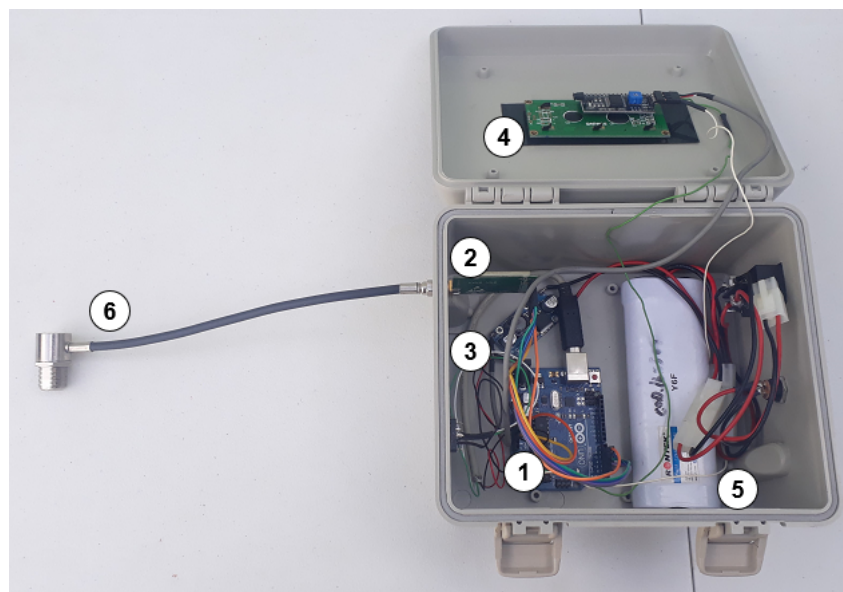


Figura 62 – Protótipo final ET - Parte interna.
Fonte Própria.

De forma similar à estação embarcada, observa-se, na Figura 62, a presença dos respectivos componentes, modelados durante a etapa de análise do domínio específico e presente na Tabela 59, etapa em que foram definidas as soluções escolhidas para composição do domínio físico, referente a parte mecânica e eletrônica. Os componentes foram listados, conforme numeração presente na ilustração, em atendimento à integração dos componentes vinculados à plataforma, conforme Figura 56, e em consonância as ligações e conexões elétricas apresentadas no Diagrama Físico da Figura 57.

1. Interface Microcontrolada - Modelo Arduino Uno;
2. Radiotransmissor - Modelo E32-433T30D;
3. Componente Eletrônico - Modelo LM2596;
4. Componente Eletrônico - Modelo Display 16x2 + I2C;
5. Bateria - Modelo SC;
6. Antena Direcional - Cabo de Conexão PigTail.

A composição externa da unidade para condicionamento dos equipamentos é apresentada na Figura 63

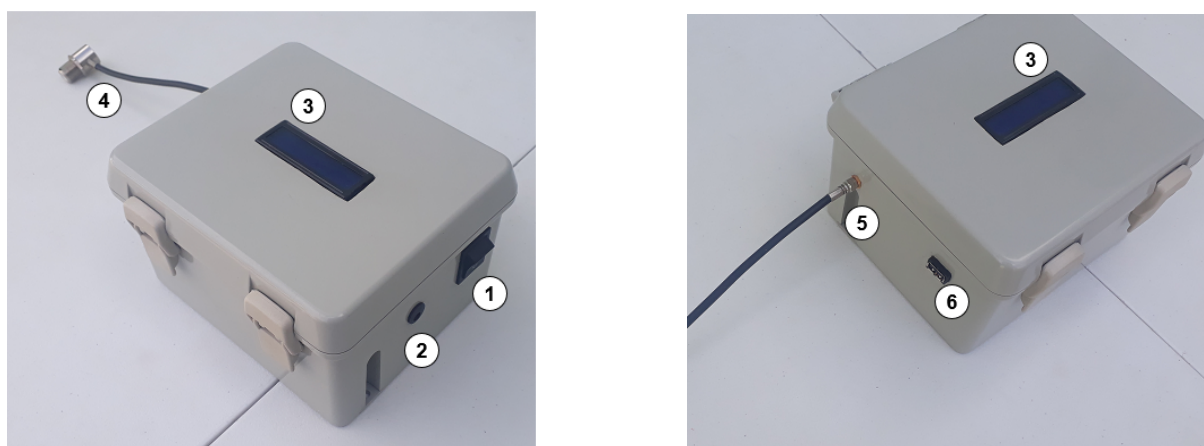


Figura 63 – Protótipo final ET - Parte externa.
Fonte Própria.

A interface externa da unidade para condicionamento, vai de encontro às necessidades listadas anteriormente. Para operação do sistema pelo usuário, as entradas e saídas aqui descritas, foram apresentadas nos diagramas de integração física, referentes à parte elétrica, Figura 57, e projetadas de acordo com as condições presentes na Figura 58. As interfaces estão listadas, a seguir, em conformidade com as numerações expostas na Figura 63.

1. Botão Liga/Desliga;
2. Conector P4 Jack;
3. Display LCD;
4. Conector PigTail - RGC58 N (Fêmea) + SMA (Macho);
5. Conector SMA (Fêmea);
6. Conector USB (Fêmea);

A próxima seção, apresenta um teste prático de operação do conjunto ET e EE, inseridas em um cenário de transmissão e recepção reais.

5.1.5 Testes práticos de comunicação

Após a conclusão da concepção do protótipo final, englobando os domínios físicos e lógicos, buscou-se realizar testes práticos para validação das funcionalidades do sistema. Entretanto, não foi possível a realização de testes dentro das condições ideais que envolvem a propagação no espaço livre, em decorrência da inexistência de missões no período compreendido entre fevereiro e maio de 2019, ou seja, após o período de construção da plataforma estratosférica. Em virtude desta situação, buscou-se validar o sistema dentro de aspectos próximos da transmissão via satélite, porém em operação horizontal, idêntica às condições aplicadas às comunicações terrestres, em que foram consideradas as seguintes circunstâncias:

- Transmissão em visada direta;
- Distância dentro dos limites validados pelo modelo virtual;
- Menor impacto de obstrução na Zona de Fresnel;

Em análise as distâncias adotadas pelo modelo virtual, o teste realizado buscou rastrear a melhor condição de visada em locais de acesso público, dentre as distâncias adotadas durante os testes do modelo virtual, realizados na Seção 5.1.2.3, correspondentes a 4, 6 e 10 Km.

Importante destacar que o cenário proposto esta inserido em uma área urbana, onde conseqüentemente existe uma grande influência de sinais aleatórios, que acarretam um aumento significativo de ruídos e interferências, além de problemas relacionados a obstruções oriundas de edificações, vegetações, rios e lagos, agentes que podem gerar operações anormais nos sistemas de comunicações terrestres. Relacionados a condições de reflexão, difração e dispersão, proporcionadas pelos tipos de materiais, ou condições atmosféricas do ambiente, que podem impactar no desempenho da transmissão.

Por outro lado, dentro do limite estabelecido para alcance da comunicação, a existência de uma transmissão com sucesso para o sistema proposto nesta pesquisa, mesmo com todas estas condições de impacto, acarreta a expectativa de maiores distâncias dentro de um cenário apropriado de comunicação via satélite, por esta ocorrer no espaço livre, e a incidência destas variáveis citadas impactarem com menor proporção.

O local escolhido para a realização dos testes práticos, foi a cidade de Goiânia, considerando uma região da metrópole com baixo índice de edificações, afim de priorizar um baixo nível de obstruções e interferências nas condições de comunicação do link.

Em virtude das informações apresentadas, iniciou-se as devidas configurações e parametrizações necessárias para a realização do teste, sendo assim as próximas seções apresentam os procedimentos realizados, considerando a ordem cronológica dos mesmos.

5.1.5.1 Análise do perímetro

A definição das condições para composição do link de comunicação, partiram inicialmente do mapeamento do perímetro, em atendimento às distâncias proposta de 4, 6 e 10 Km. Neste aspecto, foi necessário a utilização de ferramentas computacionais destinadas a análise da topografia, para um mapeamento da superfície, afim de verificar condições impeditivas que possam atrapalhar a linha de visada. Em consideração a estes fatores, o rastreamento da superfície foi realizado pelo *software Google Earth Versão Pro*, ferramenta que apoiou na geração de arquivos relacionados a topografia do terreno e espaço desejado, exportando dados específicos a respeito destas condições, os quais foram então importados para o *software Radio Mobile*, onde ocorreu a parametrização das tecnologias envolvidas referentes aos componentes radiotransmissor e antenas.

Os pontos foram devidamente mapeados pela latitude e longitude com o auxílio do *Google Earth* e estão apresentados na Figura 64, devidamente renomeados com as respectivas distâncias em relação a Estação Terrestre.

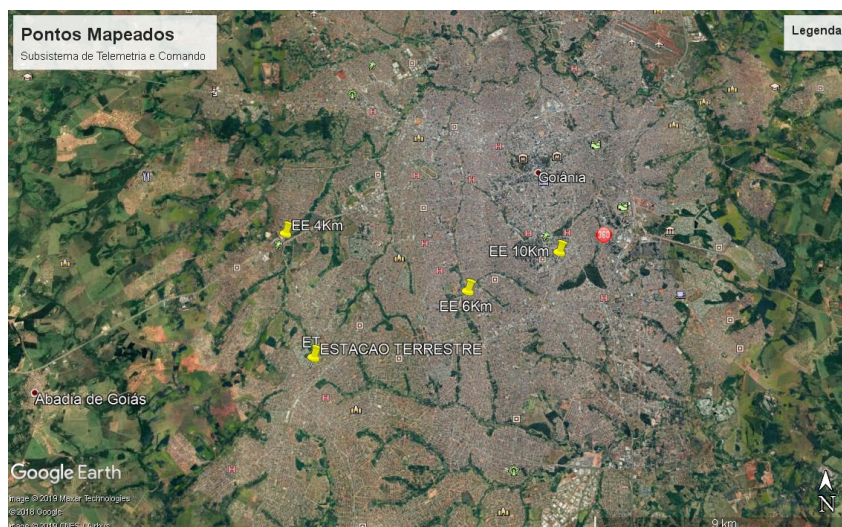


Figura 64 – Pontos mapeados.
Fonte Própria.

O mapa, gerado no *Google Earth*, foi então exportado como um arquivo com extensão kml, o qual foi importado para o *software Radio Mobile*. Com o terreno, envolvendo os possíveis pontos de comunicação mapeados, foi realizada a parametrização do radiotransmissor e da antena, para cada um dos pontos marcados em amarelo, incluindo a estação terrestre. Vale destacar que a configuração do radiotransmissor é similar para os

pontos que representam as estações embarcadas, ocorrendo alterações somente na posição da altura da antena, conforme apresentado na Figura 65.

The figure displays four screenshots of a software configuration interface, arranged in a 2x2 grid. Each window represents the configuration for a different type of radio station. The parameters shown in each window are:

- System name:** ESTACAO TERRESTRE (top-left), ESTACAO EMBARCADA 4Km (top-right), ESTACAO EMBARCADA 6Km (bottom-left), ESTACAO EMBARCADA 10Km (bottom-right).
- Transmit power (Watt):** 1 (all windows).
- Receiver threshold (µV):** 0,01 (all windows).
- Line loss (dB):** 0,5 (all windows).
- Antenna type:** yagi.ant (top-left), omni.ant (top-right, bottom-left, bottom-right).
- Antenna gain (dBi):** 10 (top-left), 3 (top-right, bottom-left, bottom-right).
- Antenna height (m):** 8 (top-left), 10 (top-right), 60 (bottom-left), 40 (bottom-right).
- Additional cable loss (dB/m):** 0 (all windows).

Buttons at the bottom of each window include "Add to Radiosys.dat" and "Remove from Radiosys.dat".

Figura 65 – Parametrização dos componentes físicos.
Fonte Própria.

O preenchimento das variáveis referentes os quadros da Figura 65, seguem os valores impostos na Tabela 60, modelados durante a configuração das variáveis do modelo virtual, para parametrização do *software Radio Mobile*. Os valores preenchidos seguem os dados das variáveis: Pulso de Entrada (Tx), que corresponde a Potência de transmissão em dBm, Ganho da Antena Tx, para a estação terrestre e Ganho da Antena Rx para as estações embarcadas. Os tipos das antenas, presentes nesta configuração, fazem referência ao modelo escolhido durante a modelagem do domínio físico específico e são configuradas conforme o modelo adotado nesta pesquisa, de acordo com o respectivo padrão de antena, sendo a Tabela 56 para a antena omnidirecional e a Tabela 57 para a antena direcional.

Uma variável que não foi discutida na modelagem deste estudo, devido à condição de comunicação em movimento, imposta pela subida e descida do satélite, diz respeito à altura da posição da antena. No caso em específico, como o teste proposto é de uma comunicação terrestre fixa, é importante o preenchimento da altura desta posição em relação ao solo, as quais foram configuradas seguindo exatamente o local, em termos da altura em metros, onde se pretende realizar a comunicação de dados, seguindo os valores adotados conforme parâmetros informados.

A próxima seção analisa a diretividade das antenas vinculadas aos pontos marcados em relação a antena da estação terrestre, considerando o mapeamento da topografia da região.

5.1.5.2 Análise da diretividade

A diretividade esta relacionada à área de visada direta entre duas antenas. A simulação do espectro de diretividade entre as antenas, é denominado *zona de Fresnel*, apresentada na Seção 2.3.3, a qual define um elipsoide de revolução, em que objetos situados dentro da primeira zona de Fresnel afetam a transmissão e causam desvios no modelo de propagação.

A simulação do comportamento da zona de Fresnel, dentro das distâncias estipuladas, visando a escolha da melhor posição para realização dos testes, foi então simulada via *software Radio Mobile*, gerando os resultados que estão presentes na Figura 66.

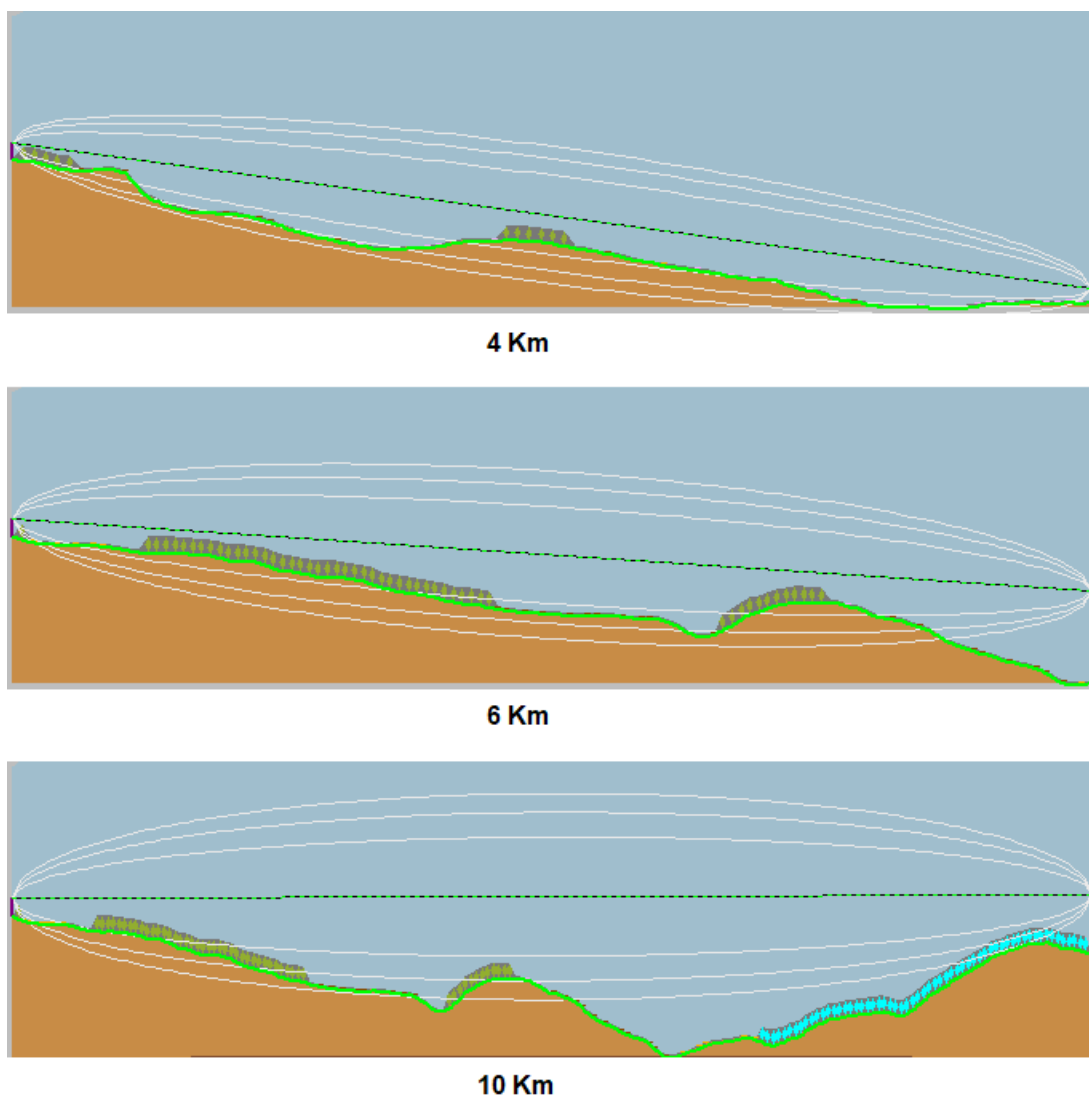


Figura 66 – Simulação das Zonas de Fresnel.
Fonte Própria.

Conforme já mencionado, a propagação em um ambiente ideal depende do caminho de visada direta entre o transmissor e o receptor, além de uma área desprovida de obstáculos ao longo do caminho. Dentro deste aspecto, e considerando o mínimo de objetos inserido na região da primeira zona de Fresnel, que possam afetar a transmissão acarretando em desvios do modelo de propagação, a Figura 66, apresenta as melhores condições dentro do contexto citado para a distância de 10 Km, a qual apresenta uma linha de visada bastante direcional, com poucos impactos provenientes de obstáculos em relação as demais distâncias propostas.

Realizando o processo inverso de exportação dos arquivos gerados, pela simulação na ferramenta computacional *Radio Mobile* em relação ao resultado apresentado para a distância de 10 Km e, em seguida importando para o *software Google Earth*, obtém-se a visualização do link de comunicação proposto, inserido no mapeamento urbano analisado, conforme apresentado na Figura 67.

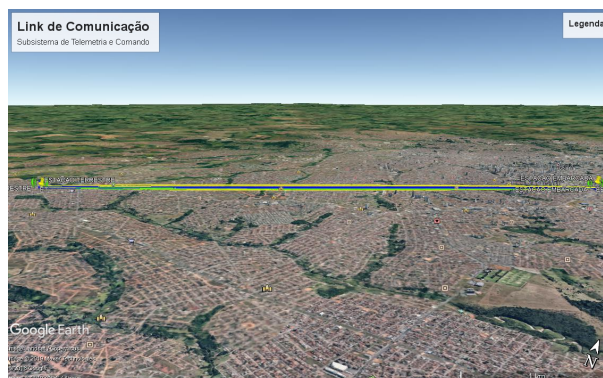


Figura 67 – Link de comunicação proposto.
Fonte Própria.

A partir do link de comunicação proposto na Figura 67, pode-se detalhar a posição das respectivas estações, vista da localização de instalação das antenas, conforme ilustra a Figura 68.



Figura 68 – Link de comunicação - Vistas detalhadas.
Fonte Própria.

Os padrões de radiação definidos em *software*, auxiliam no apontamento das

respectivas antenas, por meio do ângulo de elevação e do valor de azimute, responsável pela abertura angular do sistema de coordenadas horizontal, condição que exige uma polarização no sentido horizontal das respectivas antenas. Estas condições são apresentadas na Figura 69 e auxiliam no apontamento das antenas durante a preparação do link de comunicação.

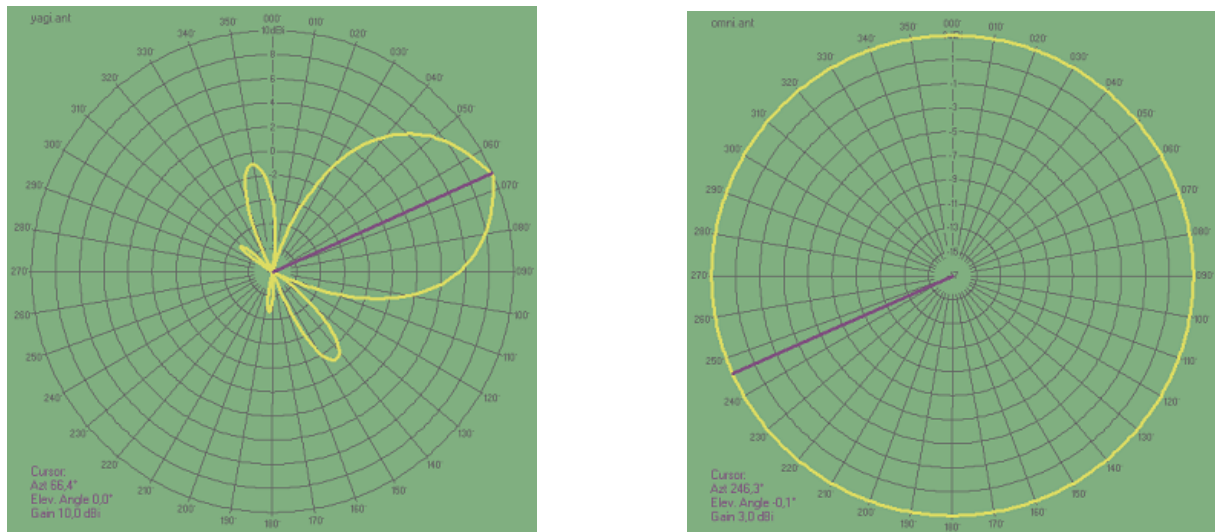


Figura 69 – Padrão de radiação das antenas.

Fonte Própria.

Em conformidade com as condições de radiação, o espectro à esquerda, apresentado na Figura 69, indica a antena direcional a ser instalada na ET e o espectro da direita, representa o espectro omnidirecional da antena posicionada no lado da EE.

Definido os pontos para operação do link de comunicação, deu-se sequência a configuração e instalação dos componentes físicos nas respectivas estações. A próxima seção analisa as condições de operação e os resultados práticos alcançados.

5.1.5.3 Análise prática

A análise prática iniciou-se com a parametrização dos radiotransmissores, tratando da configuração das variáveis de operação, relacionadas a frequência, potência, canal, paridade e taxa de transmissão. Esta atividade foi realizada via notebook, conectado pela porta USB, com o conversor USB TTL, componente eletrônico mapeado no domínio físico, responsável por realizar a interface de comunicação do radiotransmissor com o notebook, conforme ligação disponível nas Figuras 54 e 57. Sua presença nas duas estações, se faz necessária, pois a configuração deve ser feita para ambos equipamentos radiotransmissores instalados. A Figura 70 apresenta esta conexão realizada preliminarmente.

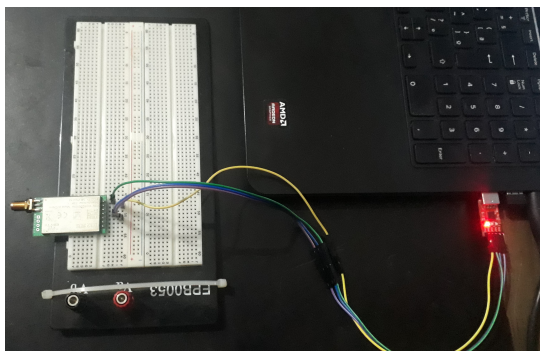


Figura 70 – Conversor USB-TTL / Radiotransmissor.
Fonte Própria.

A parametrização é realizada por meio do *software* do fabricante E-Byte, denominado *RF Setting Versão 0.70*, cuja ferramenta permite a comunicação com o dispositivo, a partir da porta serial, e a adequação dos parâmetros citados para os valores adotados no projeto. A tela apresentada na Figura 71, ilustra a configuração realizada para os dois dispositivos, vale ressaltar que, por se tratar de uma comunicação do tipo *half-duplex*, ambos os radiotransmissores devem, obrigatoriamente, possuir as mesmas configurações.

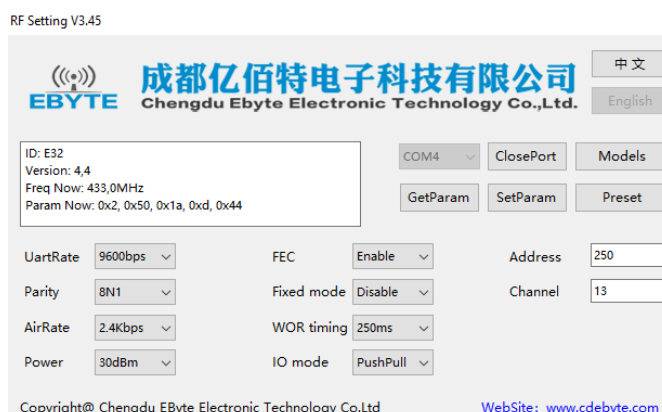


Figura 71 – Parametrização do Radiotransmissor.
Fonte Própria.

Em análise a parametrização da Figura 71, as variáveis relacionadas a frequência e potência foram adequadas pelo usuário, conforme valores atribuídos na Tabela 60. Os demais parâmetros seguiram as configurações padrões definidas pelo fabricante.

Após a etapa de configuração dos radiotransmissores, foi realizada a instalação da estação embarcada no ponto mapeado, representado pela latitude $16^{\circ}43'9,5''S$ e longitude $49^{\circ}15'35,5''O$, coordenadas geográficas do ponto em questão. Os componentes da EE foram então deixados em operação, com a antena direcionada para a posição definida pelo padrão de radiação indicado na Figura 69.

Afim de verificar a correta operação da EE, foi realizado a conexão do notebook a este subsistema, para análise via porta serial dos aspectos de transmissão. Para isto,

utilizou-se o *software PuTTY Versão 3.45*, responsável pela emulação de um terminal serial. Vale destacar que não foi desenvolvido, neste estudo, uma interface visual e amigável de interação, preocupando apenas em fornecer condições mínimas de compreensão ao usuário em relação ao funcionamento do sistema, através de linhas de informação geradas pelo código embarcado. Seguindo este entendimento, a Figura 72 expõe a condição de inicialização da operação para a EE.

```
Início da comunicação
-----
Model no.: 32
Version : 44
Features : 1E

Mode (HEX/DEC/BIN): C0/192/11000000
AddH (HEX/DEC/BIN): 2/2/10
AddL (HEX/DEC/BIN): 50/80/1010000
Sped (HEX/DEC/BIN): 1A/26/11010
Chan (HEX/DEC/BIN): D/13/1101
Optn (HEX/DEC/BIN): 44/68/1000100
Addr (HEX/DEC/BIN): 250/592/1001010000

SpeedParityBit (HEX/DEC/BIN) : 0/0/0
SpeedUARTDataRate (HEX/DEC/BIN) : 3/3/11
SpeedAirDataRate (HEX/DEC/BIN) : 2/2/10
OptionTrans (HEX/DEC/BIN) : 0/0/0
OptionPullup (HEX/DEC/BIN) : 1/1/1
OptionWakeup (HEX/DEC/BIN) : 0/0/0
OptionFEC (HEX/DEC/BIN) : 1/1/1
OptionPower (HEX/DEC/BIN) : 0/0/0
-----
Iniciando cartão SD...Erro no cartão ou cartão não inserido.
Pesquisando:
Pesquisando:
Pesquisando:
```

Figura 72 – Instalação e operação EE.
Fonte Própria.

Observa-se no terminal, apresentado na Figura 72, que o sistema foi inicializado, apresentando detalhes das configurações do radiotransmissor, permanecendo, em seguida, em um estado de espera, com status de pesquisando, ou seja, aguardando o envio de um pacote proveniente da ET. Este estado permanece indefinidamente, até que a ET entre em operação e ocorra o alinhamento do link de comunicação, através do emparelhamento direcional da radiação do espectro de frequência pelas antenas. Ainda nesta análise, vale destacar a falha na inicialização do cartão de memória, acusando um alerta de ausência do cartão de memória, situação contrária a realidade. Esta questão será discutida adiante e os seus motivos detalhados junto aos resultados finais deste ensaio prático.

Na sequência, foi instalada a estação terrestre, no ponto mapeado representado pela latitude 16°45'18,3"S e longitude 49°20'42,5"O. Um cabo, com comprimento máximo de 12m, foi conectado da antena direcional a Unidade de Condicionamento Funcional, a qual foi acionada para operação. Em seguida, ajustou-se a posição da antena, baseada no diagrama padrão de radiação apresentado na Figura 69, esta se encontrava devidamente instalada a uma altura média de 8m em relação ao solo, conforme ilustra a Figura 73.



Figura 73 – Posição Antena direcional ET.
Fonte Própria.

O processo de ajuste da posição da ET em relação a EE foi auxiliado pelo Display LCD localizado na UCF, que permitiu orientar o posicionamento, através do status informado, ou seja, a mensagem de texto apresentada ao usuário, possibilitou a orientação em relação ao refinamento do exato ponto de comunicação entre a ET e a EE. Vale destacar que este é um procedimento manual e que demanda tempo, até o correto direcionamento, cabendo ao usuário considerar e atender ao diagrama padrão gerado e aos tempos de resposta, configurados para 3 segundos no código fonte, fatores que auxiliam significativamente na agilidade deste processo.

As mensagens configuradas neste estudo, para apresentação no Display LCD, são apresentadas a seguir:

1. Mensagem de inicialização do sistema, apresentada sempre que o equipamento entra em funcionamento, representado pela Figura 74;



Figura 74 – Display LCD - Inicialização do sistema.
Fonte Própria.

2. Mensagem de transmissão do pacote, apresenta a sigla TX, o número do pacote a ser transmitido e a mensagem ou comando enviado, partindo da ET com destino a EE, representado pela Figura 75;



Figura 75 – Display LCD - Transmissão do pacote.
Fonte Própria.

3. Mensagem de *TimeOut* ou tempo excedido, informa ao usuário que o tempo de entrega do pacote foi excedido, e que o sistema está retransmitindo o pacote, representado pela sigla RTX. No código desenvolvido neste estudo, esta condição permanece indefinidamente, até que ocorra a confirmação do recebimento por parte da EE, representado pela Figura 76;



Figura 76 – Display LCD - *TimeOut* e retransmissão do pacote.
Fonte Própria.

4. Mensagem de confirmação de entrega, apresenta a sigla RX e o número do pacote entregue para validação. Esta condição simboliza a telemetria, ou seja, uma resposta da EE direcionada a ET, em relação a um comando originado da estação terrestre. No contexto deste estudo, o comando (*uplink*) está relacionado ao envio do pacote e a telemetria (*downlink*) à confirmação do recebimento do pacote, representado pela Figura 77;

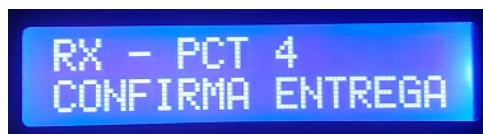


Figura 77 – Display LCD - Confirma entrega do pacote.
Fonte Própria.

O posicionamento correto do link de comunicação entre as estações se deu a partir do instante em que as mensagens de *TimeOut*, no Display LCD, foram eliminadas. Deste momento em diante, as mensagens referentes a TX e RX, passaram a se alternar concomitantemente, garantindo o estabelecimento da conexão e a correta função da atividade de Telemetria e Comando, funcionalidades almejadas nesta pesquisa.

Com o objetivo de obter uma análise mais detalhada, referente a transação dos pacotes entre as estações, conectou-se em cada lado um notebook, para o monitoramento do terminal via comunicação serial, apoiado pelo *software PuTTY*. Dessa forma, pôde-se constatar os comportamentos de transmissão e recepção para posterior comparação. Importante destacar que esta análise teve que ser considerada do lado da EE, devido a falha na inicialização do dispositivo ocasionado pelo cartão de memória, pois sua ausência inviabilizou a gravação dos dados e conseqüente armazenamento do *log* de informações para posterior análise.

A Figura 78 ilustra a instalação dos equipamentos do lado da EE e a ligação dos componentes ao notebook para monitoramento do comportamento das transações.

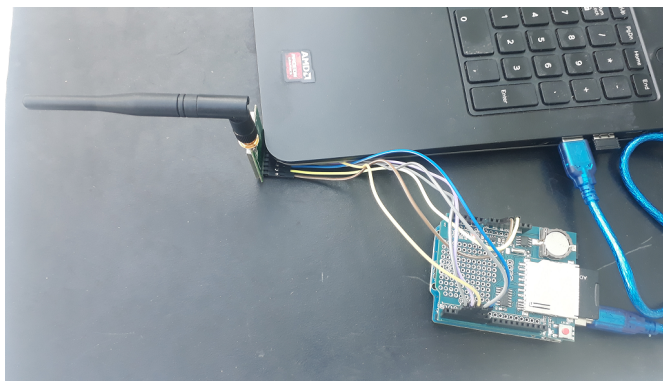


Figura 78 – Instalações EE.
Fonte Própria.

A Figura 79 ilustra a instalação dos equipamentos do lado da ET e a ligação dos componentes ao notebook para monitoramento do comportamento das transações. Detalhes sobre esta integração podem ser explorados no Apêndice E.



Figura 79 – Instalações ET.
Fonte Própria.

Um trecho comparativo da transações de pacotes, extraído deste monitoramento, é disponibilizado na Figura 80, com o lado esquerdo representado pela Estação Terrestre e o lado direito pela Estação Embarcada.

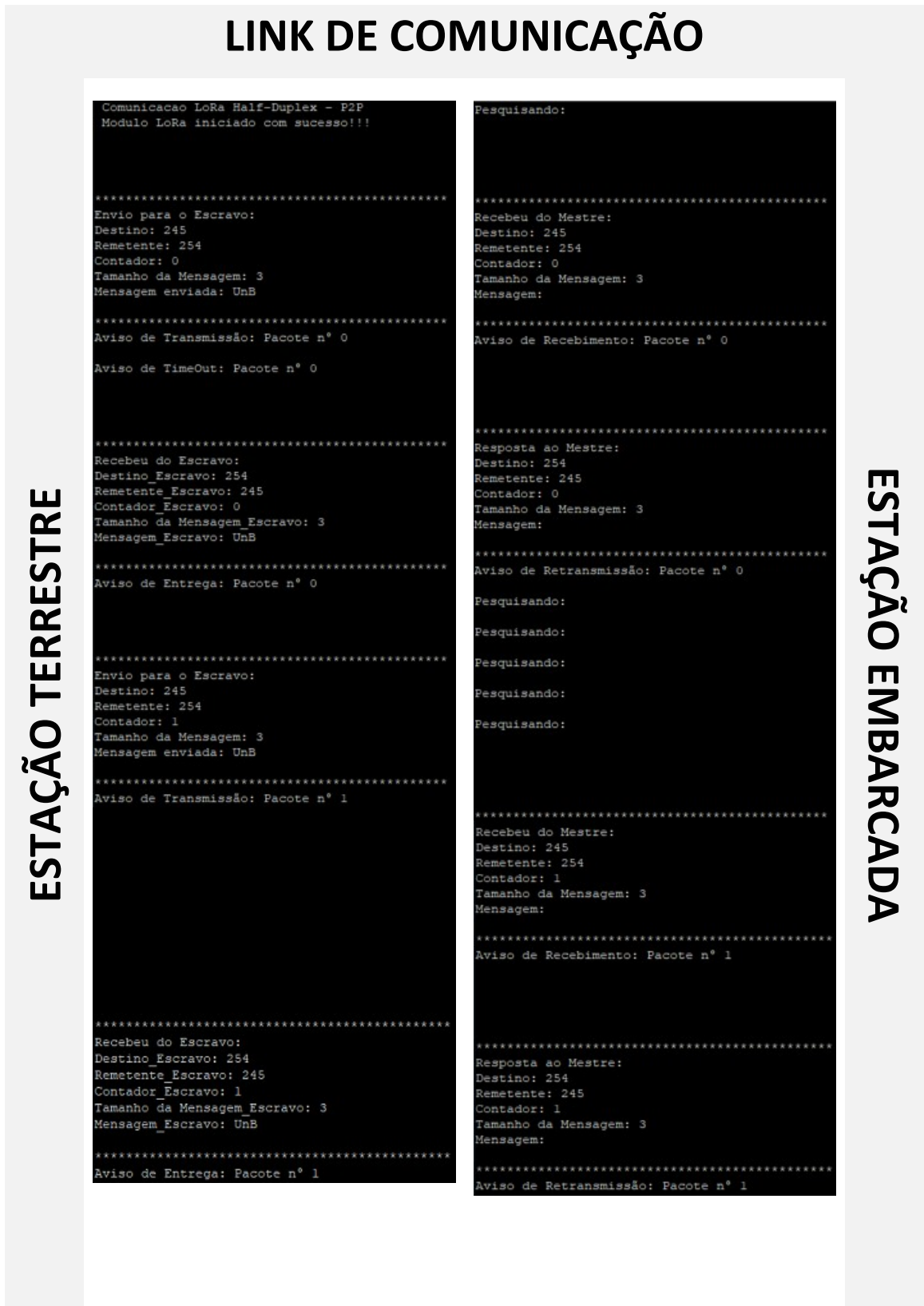


Figura 80 – Análise da transação de pacotes.
Fonte Própria.

Em análise ao trecho extraído, referente a transação de pacotes apresentada na Figura 80, pode-se observar o envio dos pacotes 0 e 1 partindo da ET, além da recepção dos respectivos pacotes do lado da EE. Após a recepção dos pacotes pela EE, os mesmos foram retransmitidos em direção a ET, sendo recebidos por esta, enviando o próximo pacote de dados. Esta condição se repete indefinidamente até que a comunicação seja interrompida em um dos lados. Caso ocorra no lado da ET, a EE ficará com status "pesquisando", aguardando uma transmissão, conforme apresentado na Figura 72. Caso a interrupção ocorra do lado da EE, a mesma entrará em uma condição de *loop* até que o pacote seja enviado e sua confirmação de entrega seja recebida pela ET. Este *loop*, com a mensagem de *TimeOut*, pode ser verificado na Figura 81.

```
Comunicacao LoRa Half-Duplex - P2P
Modulo LoRa iniciado com sucesso!!!

*****
Envio para o Escravo:
Destino: 245
Remetente: 254
Contador: 0
Tamanho da Mensagem: 3
Mensagem enviada: UnB

*****
Aviso de Transmissão: Pacote n° 0

Aviso de TimeOut: Pacote n° 0

*****
Envio para o Escravo:
Destino: 245
Remetente: 254
Contador: 0
Tamanho da Mensagem: 3
Mensagem enviada: UnB

*****
Aviso de Transmissão: Pacote n° 0

Aviso de TimeOut: Pacote n° 0
```

Figura 81 – *Loop TimeOut* - ET.
Fonte Própria.

Outra interrupção possível, considera as perdas no caminho, interferências, ruídos, ou mesmo alterações na diretividade do espectro de radiação entre a ET e a EE. Nestas condições, ambas estações permanecem em funcionamento, apresentando os status citados no parágrafo anterior, de acordo com os trechos extraídos das Figuras 72 e 81.

As análises desenvolvidas nesta seção, possibilitaram garantir os esforços dispendidos

neste estudo. A próxima seção, trás discussões de análise das especificações atendidas, ou não, no cenário de testes adotados.

5.1.6 Considerações Finais

A aplicação das técnicas proporcionados pelo MBD, atendeu aos níveis da sua Matriz de Adoção, em especial para os níveis: MBD-6, MBD-7 e MBD-8. Tais técnicas economizaram tempo de desenvolvimento, pois através da construção do modelo virtual, pôde-se entender comportamentos específicos em relação a transmissão e recepção em uma plataforma estratosférica, que até então, encontravam-se apenas no campo da teoria, baseado nas condições discutidas durante as etapas de revisão da literatura e modelagem do sistema.

A pré-concepção do modelo dentro dos ensaios, considerando as diferentes distâncias de transmissão, auxiliou na compreensão do comportamento para o espectro do sinal transmitido, esclarecendo condições de interferência e ruídos, os quais somados ao sinal desejado, podem contribuir com a redução do alcance e conseqüente prejuízo no sucesso da entrega do resultado, condições estas analisadas durante as iterações do teste MIL.

Importante salientar que a técnica MIL, contribuiu com o refinamento dos requisitos funcionais, pois abriu diversas situações referentes a condições de operação, estabilidade, controlabilidade, precisão, capacidade de transmissão e garantia de operação em tempo real. Funcionalidades que apoiaram a definição dos domínios específicos para a escolha de modelos de *hardware*, que atendessem aos parâmetros exigidos, contribuindo com a melhoria da arquitetura do sistema.

A estabilidade do comportamento de operação do modelo virtual, possibilitou no avanço do fluxo de testes, contribuindo para a geração do código originado dos blocos simulados e de suas características de funcionamento. Esta etapa de iteração de *software*, conhecida como SIL, permitiu iterações com a planta e a verificação de possíveis erros de conexões, além de incompatibilidades de *hardwares*, que ampliaram ainda mais necessidades relacionadas ao domínio lógico.

De posse do código gerado em Linguagem C, o objetivo dos testes foi viabilizar a planta, fazendo-a rodar em um ambiente de simulação, formado por um computador em tempo real com entradas e saídas, afim de garantir que o controlador "acreditasse" que estava instalado na planta real. Neste caso, a única diferença entre a aplicação final e o ambiente HIL, teste subsequente ao SIL, é a fidelidade do modelo da planta e os vetores testes utilizados.

Devido às limitações do modelo virtual, que considerou apenas os comportamentos de transmissão e recepção, ocorreu certa dificuldade de integração do teste HIL ao controlador, pois as condições propostas pelo subsistema superavam, em parte, algumas

funcionalidades não consideradas no modelo, devido a ausência dos pacotes necessários no *software*. Isto ocasionou uma limitação do modelo, quando considerado o contexto que se esperava atingir.

Esta situação foi contornada pela realização de testes diretamente no protótipo físico, considerando como requisito essencial, a aplicabilidade de componentes com baixo custo e de fácil aquisição. Assim, foi possível viabilizar, a partir dos modelos físicos propostos, a construção física do protótipo, contemplando separadamente os componentes da EE e da ET. Em atendimento às respectivas arquiteturas e conexões físicas propostas, agregou-se condições para o desenvolvimento tecnológico das interfaces contempladas por cada uma das estações, com destaque para a ET, que buscou uma identidade própria em sua concepção, em termos de integração da estrutura física, a qual foi denominada neste estudo de Unidade de Condicionamento Funcional (UCF).

Outro detalhe que exigiu adequações profundas, foi a adaptação do código gerado pelo teste SIL, em conformidade com a aplicação no protótipo. Neste contexto, condições vinculadas a bibliotecas, variáveis e determinadas estruturas condicionais foram readaptadas e refinadas para atender a arquitetura lógica proposta. Estes ajustes foram realizados por meio dos testes práticos, com o código embarcado diretamente no microcontrolador e, foram avançando progressivamente buscando a melhoria das funcionalidades do processo, em atendimento às exigências de desempenho, usabilidade, segurança e normas, destacadas nos requisitos não-funcionais.

Os testes práticos permitiram o amadurecimento e o entendimento das capacidades proporcionada pela arquitetura que envolvia as tecnologias escolhidas como solução. Neste contexto de aplicação do protótipo, dentro de um cenário real, foi possível validar as condições de Telemetria e Comando, funcionalidades que motivaram o desenvolvimento desta pesquisa.

O cenário proposto para os testes - inserido em uma área urbana com distância de 10 Km separando as estações - apesar de não ser o ideal em uma comunicação estratosférica - pois esta experimenta condições de propagação com menos interferências externas - contribuiu para resultados bastante interessantes em termos de validação das soluções aplicadas, pois foi possível concretizar, na prática, a transmissão e recepção de dados do tipo *half-duplex*, conforme meta imposta nos objetivos.

Durante os testes algumas condições impeditivas, que originaram falhas, foram observadas, as quais são listadas a seguir:

- Gravação de dados (*Log*);
- Falha na recepção da mensagem;
- Instabilidade no funcionamento do sistema.

A primeira falha identificada, está relacionada ao componente *Data Logger Shield*, presente na EE, o qual originou anomalias constantes durante o processo de gravação, não sendo possível a sua validação. Em decorrência das análises, relacionadas às substituições do cartão de memória e adequações do algoritmo, com o objetivo de otimizar e garantir um mínimo de dados provenientes da coleta durante o processo de testes, apresentaram persistência das falhas, chegando a conclusão da necessidade de troca do dispositivo para realização de novos testes. Devido ao tempo de compra e aquisição deste componente, esta substituição não ocorreu em tempo hábil para composição de resultados satisfatórios para este item.

A falha na recepção do campo mensagem do lado da EE, foi uma variável que chamou bastante a atenção, visto que as informações inseridas nos pacotes transmitidos como: endereço do destino, endereço do remetente, contador de pacotes e tamanho da mensagem, foram recebidos normalmente. Suspeita-se de uma incompatibilidade no tipo de dados manipulado entre a ET e a EE, esta questão foi testada, afim de manter as mesmas características para ambas estações, ainda assim a falha persistiu. Outra possibilidade analisada foi a questão do tamanho do pacote transmitido. Neste aspecto, foram realizadas adequações para que o pacote transmitisse apenas a mensagem, sem sucesso, em uma troca do tipo *double* para um valor inteiro, a mensagem chegou com sucesso, então faz-se necessário uma análise na transmissão de mensagens do tipo texto.

Foram observadas algumas anomalias no funcionamento do sistema em operação, o qual, em alguns momentos, exigia uma reinicialização, visando a normalização do sistema. Um tratamento mais detalhado do código e questões referentes a incompatibilidade eletromagnética devem ser analisadas, pois interferências oriundas de campos eletromagnético afetam diretamente o comportamento dos componentes eletrônicos.

Partindo de uma análise geral, este estudo permitiu validar as principais funcionalidades propostas referentes a Telemetria (*downlink*) e Comando (*uplink*) presentes em um sistema de comunicação via satélite, aplicado à plataformas estratosféricas, pois conseguiu-se modelar, testar e verificar as condições de projeto, chegando-se ao protótipo físico, que validou e viabilizou o desafio imposto no âmbito desta pesquisa.

5.2 Resumo do capítulo

O Capítulo 5 apresentou as etapas de aplicação dos testes que compõem o MBD e o cumprimento do fluxo proposto no Modelo V, apresentado na Figura 21.

Os testes seguiram a forma de análise dos métodos MIL, SIL e HIL, em seguida avançou para o protótipo físico e lógico de cada uma das estações envolvidas, chegando aos testes práticos, que proporcionaram a análise do comportamento do subsistema de Telemetria e Comando aplicada a solução proposta.

A próxima seção, diz respeito a conclusão, apresentando a contribuição desta pesquisa no âmbito acadêmico e científico, além de agregar propostas de trabalhos futuros com o intuito de promover avanços no cenário explorado.

6 Conclusões

O presente trabalho teve como objetivo principal conceber um subsistema de comunicação ponto-a-ponto, dedicado à missões espaciais para plataformas estratosféricas, atendendo ao padrão CubeSat. Neste aspecto, foi necessário propor, modelar e validar uma metodologia capaz de fornecer subsídios no auxílio às etapas de desenvolvimento do projeto.

Dado o cenário de um subsistema de comunicação, que agrega funções de Telemetria e Comando, voltado para uma plataforma estratosférica, conforme o padrão CubeSat. A integração dos modelos empregados, em conformidade com as interações entre os domínios específicos, proporcionou soluções que atenderam aos requisitos especificados.

6.1 Proposta de Integração das Metodologias

Para o desenvolvimento da concepção do subsistema pretendido, foi necessário definir as melhores metodologias, envolvendo praticidade no desenvolvimento de modelos, durante a etapa de integração dos domínios específicos, proporcionando redução nas fases de não desenvolvimento, através da combinação unificada das etapas de projeto, implementação e testes.

Em resumo, a preocupação consistiu em definir soluções em relação aos domínios específicos e validá-las através da elaboração de um modelo virtual que agregasse as funcionalidades pretendidas. Possibilitando maior controle na seleção das melhores soluções, através da realização de testes simultâneos, resultando em maior interação e confiança entre a simulação e a prototipagem, além de minimizar os custos financeiros com a aquisição de equipamentos de forma precipitada.

Ainda neste contexto, os modelos MBD e V, se mostraram bastante eficazes e a integração desses resultado em um modelo híbrido contribuiu decisivamente, pois conseguiu associar etapas do desenvolvimento a simulações virtuais, ou seja, através da adoção de integração destes modelos, pôde-se definir os requisitos importantes para o funcionamento do sistema e, posteriormente, a partir de sua descrição textual, foi possível a definição da especificação do sistema, resultando em sua arquitetura. Com o modelo virtual elaborado, aplicou-se os testes e verificações em *loop* para validação das etapas anteriores, melhoria do processo e geração do código na linguagem desejada, a qual foi refinada e embarcada na solução para o *hardware* escolhido. Finalmente, pôde-se chegar a concepção do subsistema de Telemetria e Comando.

6.2 Soluções dos Domínios Específicos

O processo de modelagem do subsistema de comunicação, iniciou-se pelo levantamento das necessidades dos clientes, permitindo o rastreio dos requisitos funcionais, relacionando as prioridades, vínculos e quantificações e dos não-funcionais, vinculados ao comportamento em termos de usabilidade, desempenho, segurança e padrões. Os requisitos foram separados quanto a abrangência de atuação dentro do sistema, relacionado ao nível de importância destes em cada cenário das estações ET e EE.

Por envolver diferentes domínios específicos, que englobam mecânica, eletrônica e *software*, houve a necessidade de um envolvimento multidisciplinar, que ampliou a coesão na interdependência entre as áreas envolvidas, além do alinhamento, das necessidades e requisitos, a um escopo viável para o que se pretendia desenvolver. Neste aspecto, a arquitetura foi distribuída, considerando os aspectos físicos e lógicos, o que possibilitou separar, inicialmente, soluções mecânicas e eletrônicas das soluções de *software*.

Os requisitos possibilitaram o rastreamento das arquiteturas físicas e lógicas, contemplando, para a primeira, componentes do tipo COTS de baixo custo e, na segunda as reais necessidades de operação do sistema para atendimento às exigências inseridas nas respectivas estações, onde a partir da definição da arquitetura para cada domínio específico, pôde-se proceder com o envolvimento dos domínios em um único bloco.

6.3 Modelo Virtual, Testes e Verificações

A compreensão sobre o que se deseja aplicar ao subsistema, propiciado pelo amadurecimento na etapa de integração dos domínios, permitiu a elaboração do modelo virtual, em atendimento ao fluxo proposto no Modelo V, apresentado na Figura 21, garantindo o atendimento aos níveis MBD-4, MBD-6, MBD-7 e MBD-8, presentes na Tabela 6, em cumprimento à Matriz de Adoção do MBD.

A construção do modelo virtual, foi baseada na similaridade dos blocos de simulação com as soluções propostas na arquitetura, o que permitiu, nesta etapa de desenvolvimento, contar com um cenário próximo das condições reais de operação. A análise do modelo virtual foi pensada em condições de transmissões dentro de distâncias que viabilizem a comprovação durante a validação pelo protótipo físico.

As técnicas adotadas utilizadas para validação seguiram a seguinte sequência: *Model-In-the-Loop* (MIL), *Software-In-the-Loop* (SIL) e *Hardware-In-the-Loop* (HIL).

O teste MIL contribuiu com o refinamento dos requisitos funcionais, pois abriu um leque de situações referentes às condições de operação, estabilidade, controlabilidade, precisão, capacidade de transmissão e garantia de operação em tempo real, funcionalidades que apoiaram na definição dos domínios específicos para a escolha de modelos de *hardware*

que atendessem aos parâmetros exigidos, contribuindo com a melhoria da arquitetura do sistema.

O teste SIL permitiu iterações com a planta e a verificação de possíveis erros de conexões, além de incompatibilidades de *hardwares* que ampliaram ainda mais necessidades relacionadas ao domínio lógico.

Devido às limitações do modelo virtual, que considerou apenas os comportamentos de transmissão e recepção, ocorreu certa dificuldade de integração do teste HIL ao controlador, visto que as condições propostas pelo subsistema superaram, em parte, algumas funcionalidades, não consideradas no modelo, devido a ausência dos pacotes necessários no *software Simulink*[®], limitando o modelo.

6.4 Concepção do Protótipo

A concepção do protótipo final atendeu às expectativas impostas pelas arquiteturas, tanto físicas quanto lógicas, além do atendimento às conexões e ligações vinculadas ao diagrama físico elaborado, agregando resultados finais satisfatórios no desenvolvimento tecnológico das interfaces contempladas por cada uma das estações. Destaque para a ET, que apresentou uma identidade própria, resultando em um produto específico, que conciliou integração, usabilidade, flexibilidade e segurança em sua estrutura física, denominada de Unidade de Condicionamento Funcional da Estação Terrestre (UCF-ET), nome dado pela composição formada por um único bloco, que condiciona os componentes em cumprimento aos requisitos funcionais englobados pela estação terrestre.

Os testes práticos realizados, foram inseridos em um contexto similar ao realizado na fase de simulação, em consideração ao contexto da distância de transmissão. Entretanto, por ter sido realizado em um perímetro urbano, experimentou condições de propagação com alto índice de interferência e ruído, o que, apesar de não experimentar condições ideais para um subsistema de comunicação via satélite, favoreceu em relação aos resultados obtidos, devido à realização, com sucesso, das funcionalidades de Telemetria e Comando durante a operação do subsistema. Isto possibilitou, na prática, a transmissão e recepção de dados do tipo *half-duplex*, o que é bastante importante, pois deixa a expectativa da possibilidade de alcances mais amplos, para aplicações em cenários de comunicações reais, para os quais o subsistema foi concebido.

6.5 Trabalhos Futuros

Esta pesquisa buscou agregar contribuições no âmbito da pesquisa científica e acadêmica, que envolvem diferentes domínios da engenharia, com destaque para: teleco-

municações, produção, computação, *software*, eletrônica, mecânica além de realizar um apanhado geral à integração de sistemas, trazendo para o contexto da mecatrônica.

Por relacionar diferentes domínios, faz-se necessário algumas ponderações em relação a futuros trabalhos que possam contribuir, ou dar continuidade, proporcionando o crescimento deste estudo. Estas, são apresentadas a seguir e em continuidade a lógica imposta no texto, são distribuídas, como propostas, para os respectivos domínios, físicos e lógicos.

A lista a seguir apresenta propostas levantadas para trabalhos futuros no domínio físico:

- Ampliação da capacidade de portas da interface microcontrolada, visando um aumento na coleta de dados para Telemetria, provenientes de equipamentos e sensores presentes na plataforma embarcada;
- Ampliação da capacidade processamento e controle da interface microcontrolada;
- Aplicação de um sistema de comunicação *Full-Duplex*, permitindo a transmissão e recepção simultânea, trabalhando em frequências diferentes de propagação de sinal;
- Aprimoramento no sistema de antenas, adotando um arranjo de antenas adaptativas na estação terrestre e, antenas com melhores condições de ganho e diretividade na estação embarcada;
- Estudo da compatibilidade eletromagnética, aplicada principalmente à estação embarcada, por se tratar de uma plataforma bastante reduzida e que transporta diversos tipos de equipamentos com diferentes comportamentos de operação;
- Ampliação das condições para melhor desempenho do radiotransmissor, com inserção de equipamentos para amplificação de sinais, codificadores/decodificadores; multiplexadores/demultiplexadores, filtros, entre outros, contribuindo para melhores resultados na transmissão e recepção dos dados.

A lista a seguir apresenta propostas levantadas para trabalhos futuros no domínio lógico:

- Aprimoramento do sistema computacional, buscando uma interface amigável e intuitiva, estrutura para tratamento de erros e parametrização de comandos;
- Integração de protocolo de comunicação, buscando compatibilidade com outros sistemas;
- Aplicação de técnicas de processamento digital de sinais, visando a manipulação de dados, como imagens, vídeos e voz;

- Ampliação da capacidade de armazenamento de dados, principalmente na estação embarcada;
- Sistema para registro de *Log* de informações eficiente.

Importante salientar que o estudo atendeu às metas estabelecidas em seus objetivos, geral e específicos. Uma grande contribuição a ser destacada, consiste na condição de repetibilidade da metodologia, proporcionada pelo arranjo das etapas em composição as fases impostas pelo modelo híbrido, considerando aspectos do MBD e V, auxiliando em futuras replicações deste sistema, independente dos tipos de soluções envolvidas.

Referências

- AARENSTRUP, R. Managing model-based design. the mathworks. *Inc.: Natick, MA, USA*, 2015. Citado na página 47.
- AGASID, E. et al. Small spacecraft technology state of the art. *NASA, Ames Research Center, Mission Design Division Rept. NASA/TP-2015-216648/REV1, Moffett Field, CA*, 2015. Citado 6 vezes nas páginas 8, 9, 12, 29, 30 e 31.
- ANATEL. *Resolução nº 671, de 3 de novembro de 2016*. <http://www.anatel.gov.br>, 2016. Resolução nº 671, de 3 de novembro de 2016. Disponível em: Legislação - Resoluções – 2016 - Resolução nº 671, de 3 de novembro de 2016. Citado na página 32.
- ANATEL. *Ato nº 9106, de 22 de novembro de 2018*. <http://www.anatel.gov.br>, 2018. Ato nº 9106, de 22 de novembro de 2018. Disponível em: Legislação – 2018 - Atos de Requisitos Técnicos de Gestão do Espectro -Resolução nº 697, de 28 de agosto de 2018. Citado na página 33.
- ANATEL. *Plano de Atribuição, Destinação e Distribuição de Faixas de Frequências no Brasil*. <http://www.anatel.gov.br>, 2018. Resolução nº 697, de 28 de agosto de 2018. Disponível em: Resoluções – Acervo Documental – Planos – de atribuição de faixas de frequências. Citado na página 30.
- ANATEL. *Resolução nº 697, de 28 de agosto de 2018*. <http://www.anatel.gov.br>, 2018. Resolução nº 697, de 28 de agosto de 2018. Disponível em: Resoluções – 2018 - Resolução nº 697, de 28 de agosto de 2018. Citado 2 vezes nas páginas 31 e 35.
- ARGOS. *Argo's User's Manual*. 2016. Acessado em: 01/05/2019. Disponível em: <http://www.argos-system.org/manual/index.html#3-location/32_principle.htm>. Citado na página 19.
- BACK, N. et al. Projeto integrado de produtos: planejamento, concepção e modelagem. *Barueri: Malone*, p. 435–482, 2008. Citado na página 36.
- BALANIS, C. *Teoria de Antena: Análise e Síntese. 3ª*. [S.l.]: Ed., Rio de Janeiro, LTC, 2009. Citado 2 vezes nas páginas 26 e 29.
- BARBALHO, S. C. Modelo de referência para o desenvolvimento de produtos mecatrônicos: proposta e aplicações. *USP, São Carlos*, p. 275, 2006. Citado na página 45.
- BARBIERI, G.; FANTUZZI, C.; BORSARI, R. A model-based design methodology for the development of mechatronic systems. *Mechatronics*, Elsevier, v. 24, n. 7, p. 833–843, 2014. Citado na página 36.
- BEKKER, D. et al. The cove payload—a reconfigurable fpga-based processor for cubesats. 2011. Citado na página 13.
- BERNARDI, M.; BLEY, H.; SCHMITT, B. New approaches for developing mechatronical products in multidisciplinary teamwork. In: *The 35th CIRP-International Seminar on Manufacturing Systems*. [S.l.: s.n.], 2002. p. 13–15. Citado na página 45.

- BERTIGER, B. R.; LEOPOLD, R. J.; PETERSON, K. M. *Telemetry, tracking and control for satellite cellular communication systems*. [S.l.]: Google Patents, 1993. US Patent 5,187,805 Acessado em: 10/04/2019. Citado na página 15.
- BETTIG, B.; GERSHENSON, J. K. The representation of module interfaces. *International Journal of Product Development*, Inderscience Publishers, v. 10, n. 4, p. 291–317, 2010. Citado na página 41.
- BHISE, V. D. *Designing complex products with systems engineering processes and techniques*. [S.l.]: CRC Press, 2013. Citado na página 36.
- BLANDFORD, R. D. et al. New worlds, new horizons in astronomy and astrophysics. *National Research Council, The National Academies Press, Washington, DC*, 2010. Citado na página 5.
- BOARD, S. S.; COUNCIL, N. R. et al. *Earth science and applications from space: National imperatives for the next decade and beyond*. [S.l.]: National Academies Press, 2007. Citado na página 5.
- BOARD, S. S.; COUNCIL, N. R. et al. *Vision and voyages for planetary science in the decade 2013-2022*. [S.l.]: National Academies Press, 2012. Citado na página 5.
- BORGES, R. A. et al. Laicansat-5: A mission for recording the total solar eclipse from the stratosphere. In: IEEE. *2018 IEEE Aerospace Conference*. [S.l.], 2018. p. 1–7. Citado 2 vezes nas páginas 1 e 36.
- B.OSBORNE. *QB50 UNSW EC0*. 2015. Acessado em: 15/11/2018. Disponível em: <<http://www.acser-archive.unsw.edu.au/downloads/2015Cubesat/19-Osborne.pdf>>. Citado na página 9.
- BOUWMEESTER, J.; GUO, J. Survey of worldwide pico-and nanosatellite missions, distributions and subsystem technology. *Acta Astronautica*, Elsevier, v. 67, n. 7-8, p. 854–862, 2010. Citado 2 vezes nas páginas 7 e 14.
- BRASIL. *Portaria N° 307, de 22 de julho de 2009*. 2009. Norma de ativação e execução dos serviços da Rede Nacional de Emergência de Radioamadores - RENER. Citado na página 34.
- BRUDER, J. et al. Ieee standard for letter designations for radar-frequency bands. *IEEE Aerospace & Electronic Systems Society*, p. 1–3, 2003. Citado na página 29.
- BUUR, J. *A theoretical approach to mechatronics design*. [S.l.]: Institute for Engineering Design, Technical University of Denmark Denmark, 1990. Citado na página 46.
- CGEE. *CubeSats Resumo executivo*. 2016. Centro de Gestão e Estudos Estratégicos. Acessado em: 12/11/2018. Disponível em: <<http://tpu.ru/en/news-events/788/>>. Citado 3 vezes nas páginas 1, 7 e 8.
- CHO, M.; HIROKAZU, M.; GRAZIANI, F. Introduction to lean satellite and iso standard for lean satellite. In: IEEE. *2015 7th International Conference on Recent Advances in Space Technologies (RAST)*. [S.l.], 2015. p. 789–792. Citado 2 vezes nas páginas 1 e 6.

- CURRAN, R. et al. A methodology for mechatronic products design applied to the development of a instrument for soil compaction measurement. *ary Lif System*, p. 396, 2015. Citado 2 vezes nas páginas 60 e 63.
- DIAZ-AGUADO, M. et al. Small class-d spacecraft thermal design, test and analysis - pharماسat biological experiment. 03 2009. Citado na página 14.
- DUBOCK, P. et al. The envisat satellite and its integration. *ESA bulletin*, v. 106, p. 26–45, 2001. Citado na página 5.
- EBYTE. *E32 Series User Manual - SX1276/SX1278 Wireless Module*. 2017. Acessado em: 05/06/2019. Disponível em: <www.ebyte.com/en/product-view-news.aspx?id=108>. Citado na página 120.
- GAUSEMEIER, J.; MOEHRINGER, S. Vdi 2206-a new guideline for the design of mechatronic systems. *IFAC Proceedings Volumes*, Elsevier, v. 35, n. 2, p. 785–790, 2002. Citado 5 vezes nas páginas 2, 3, 37, 38 e 43.
- GOBER. *Antena Yagi UHF*. 2018. Acessado em: 19/06/2019. Disponível em: <<http://www.gober.com.br/produtos.asp>>. Citado na página 101.
- GUEST, A. N. Telemetry, tracking, and command (tt&c). *Handbook of Satellite Applications*, Springer, p. 1–12, 2016. Citado 6 vezes nas páginas 15, 16, 17, 18, 19 e 20.
- HAYKIN, S.; MOHER, M. *Introdução aos sistemas de comunicação*. [S.l.]: Bookman Editora, 2009. Citado 6 vezes nas páginas 21, 22, 23, 24, 25 e 28.
- HEHENBERGER, P.; EGYED, A.; ZEMAN, K. Understanding the relationship of information in mechatronic design modeling. In: . [S.l.: s.n.], 2011. p. 113–120. Citado 4 vezes nas páginas 2, 37, 38 e 39.
- HEHENBERGER, P. et al. Hierarchical design models in the mechatronic product development process of synchronous machines. *Mechatronics*, v. 20, p. 864–875, 12 2010. Citado 4 vezes nas páginas 36, 43, 44 e 50.
- HEHENBERGER, P.; ZEMAN, K. Design activities in the development process of mechatronic systems. In: . [S.l.: s.n.], 2007. p. 1 – 6. ISBN 978-1-4244-1264-8. Citado 4 vezes nas páginas 2, 37, 38 e 42.
- IBM. *Conceito: Arquitetura de Sistema*. 2006. Acessado em: 15/06/2019. Disponível em: <http://mds.cultura.gov.br/core.base_rup/guidances/concepts/system_architecture_5F3B1E17.html>. Citado na página 112.
- INITIATIVE, N. C. L. et al. *CubeSat 101: Basic Concepts and Processes for First-Time CubeSat Developers*. [S.l.], 2017. Citado na página 9.
- ISIS. *ISIS on board computer*. 2016. Acessado em: 10/11/2018. Disponível em: <<http://www.isispace.nl/product/on-boardcomputer/>>. Citado na página 14.
- IVANCIC, W. D. Architecture and system engineering development study of space-based satellite networks for nasa missions. 2003. Citado na página 17.
- JOHNSON, N. L.; STANSBERY, E. G. The new nasa orbital debris mitigation procedural requirements and standards. *Acta Astronautica*, Elsevier, v. 66, n. 3-4, p. 362–367, 2010. Citado na página 12.

- JPL, N. J. P. L. *Cassini Solstice Mission*. 2016. Acessado em: 10/04/2019. Disponível em: <<http://saturn.jpl.nasa.gov/mission/quickfacts/>>. Citado na página 5.
- KARA, O. et al. Communication architecture and international policy recommendations enabling the development of global cubesat space networks. In: *66th International Astronautical Congress*. [S.l.: s.n.], 2015. Citado na página 31.
- KEESEEE, C. *Satellite Telemetry, Traking, and Control Subsystems*. 2003. Acessado em: 10/02/2019. Disponível em: <http://ocw.mit.edu/courses/aeronautics-and-astronautics/16-851-satelliteengineering-fall-2003/lecture-notes/l20_satellitettc.pdf>. Citado 2 vezes nas páginas 18 e 20.
- KILGER, C. et al. *Supply chain management and advanced planning*. [S.l.]: Springer, 2015. Citado na página 36.
- KLOFAS, B.; ANDERSON, J.; LEVEQUE, K. A survey of cubesat communication systems. In: *5th Annual CubeSat Developers' Workshop*. [S.l.: s.n.], 2008. Citado na página 35.
- KOMOTO, H.; TOMIYAMA, T. A framework for computer-aided conceptual design and its application to system architecting of mechatronics products. *Computer-Aided Design*, v. 44, p. 931–946, 10 2012. Citado na página 43.
- KOSSIAKOFF, A.; SWEET, W. N. et al. *Systems engineering: Principles and practices*. [S.l.]: Wiley Online Library, 2003. Citado na página 36.
- LARSON, W. J.; WERTZ, J. R. *Space mission analysis and design*. [S.l.], 1992. Citado 4 vezes nas páginas 12, 13, 14 e 18.
- LEWIS, B. S. et al. Biosentinel: Monitoring dna damage repair beyond low earth orbit on a 6u nanosatellite. 2014. Citado na página 14.
- LEY, W.; WITTMANN, K.; HALLMANN, W. *Handbook of space technology*. [S.l.]: John Wiley & Sons, 2009. v. 22. Citado na página 11.
- MALONEY, P.; NURSILO, W. Optimizing performance and fuel economy of a dual-clutch transmission powertrain with model-based design. *Optimizing Performance and Fuel Economy of a Dual-Clutch Transmission Powertrain with Model-Based Design*, 2011. Citado na página 46.
- MARTINEZ, A.; PETRO, A. Optical communications and sensor demonstration. 2015. Citado 2 vezes nas páginas 13 e 14.
- MARTÍNEZ, J. L. *Zonas de Fresnel en un radioenlace*. 2018. PRORED. Acessado em 09/04/2019. Disponível em: <<https://www.prored.es/blog/radio-enlace/zonas-de-fresnel-en-un-radioenlace/>>. Citado na página 28.
- MASON, J. et al. Minxss cubesat on-orbit performance and the first flight of the blue canyon technologies xact 3-axis adcs. 2016. Citado na página 13.
- MFHRUG. *Sistemas Simplex, Half-Duplex e Full-Duplex*. 2012. Acessado em: 15/04/2019. Disponível em: <<http://en.federalspace.ru/174/>>. Citado na página 22.

- MHENNI, F. et al. A sysml-based methodology for mechatronic systems architectural design. *Advanced Engineering Informatics*, v. 28, 08 2014. Citado 4 vezes nas páginas 2, 37, 38 e 41.
- MLAMBO, P. et al. Methodologies for mechatronic systems design: Attributes and popularity. v. 12, p. 123–133, 05 2018. Citado 2 vezes nas páginas 42 e 43.
- MSU. *PrintSat: Mission Overview*. 2015. Acessado em: 20/04/2019. Disponível em: <<https://ssel.montana.edu/printsat.html>>. Citado na página 9.
- MUELLER, J.; HOFER, R.; ZIEMER, J. Survey of propulsion technologies applicable to cubesats. Pasadena, CA: Jet Propulsion Laboratory, National Aeronautics and Space . . . , 2010. Citado na página 12.
- NASA. *Heliophysics*. 2015. Acessado em: 10/04/2019. Disponível em: <<http://science.nasa.gov/heliophysics/>>. Citado na página 5.
- NASA. *NASA Space Communication: TDRSS*. 2019. Acessado em: 20/03/2018. Disponível em: <<https://www.nasa.gov/directorates/heo/scan/index.html>>. Citado na página 18.
- NASA, N. *Systems Engineering Handbook*. Vol. [S.l.], 2007. Citado na página 36.
- OPENCADD, U. *Ramp-Up de Model-Based Design*. 2019. <https://opencadduniversity.com.br/home/course/ramp-up-de-model-based-design/2>. Acessado em: 10/05/2019. Disponível em: <<https://ssel.montana.edu/printsat.html>>. Citado 4 vezes nas páginas 47, 50, 56 e 57.
- PELTON, J. N.; MADRY, S.; LARA, S. C. Satellite applications handbook: The complete guide to satellite communications, remote sensing, navigation, and meteorology. *Handbook of Satellite Applications*, Springer, p. 1–17, 2016. Citado na página 21.
- PIETRUSEWICZ, K. Metamodelling for design of mechatronic and cyber-physical systems. *Applied Sciences*, Multidisciplinary Digital Publishing Institute, v. 9, n. 3, p. 376, 2019. Citado 5 vezes nas páginas 2, 37, 38, 46 e 47.
- PIGNATELLI, D.; MEHRPARVAR, A. Cubesat design specifications rev. 13. *The CubeSat Program, Cal Poly SLO*, California Polytechnic State Univ. San Luis Obispo, CA, 2013. Citado 3 vezes nas páginas 1, 7 e 36.
- PISACANE, V. L. *Fundamentals of space systems*. [S.l.]: Johns Hopkins University/Appli, 2005. Citado na página 17.
- POGHOSYAN, A.; GOLKAR, A. Cubesat evolution: Analyzing cubesat capabilities for conducting science missions. *Progress in Aerospace Sciences*, Elsevier, v. 88, p. 59–83, 2017. Citado 8 vezes nas páginas 1, 7, 8, 9, 10, 12, 13 e 14.
- PRESSMAN, R. S. *Engenharia de Software*. 5 a edição. Ed. [S.l.]: Makron Books, 2002. Citado na página 108.
- PRIES, K. H.; QUIGLEY, J. M. *Project management of complex and embedded systems: Ensuring product integrity and program quality*. [S.l.]: Auerbach Publications, 2008. Citado na página 36.

- PUGH, S. Total design: integrated methods for successful product engineering. Addison-Wesley Wokingham, 1991. Citado na página 36.
- PUIG-SUARI, J.; TURNER, C.; TWIGGS, R. Cubesat: the development and launch support infrastructure for eighteen different satellite customers on one launch. 2001. Citado na página 7.
- PUTMAN, P. et al. Cryogenic thermal management for cryocube-1. 2015. Citado na página 15.
- RAPPAPORT, T. S. *Comunicações sem fio: princípios e práticas*. [S.l.]: Pearson Prentice Hall, 2009. Citado 4 vezes nas páginas 23, 24, 25 e 27.
- REUTLINGER, A. et al. Fiber optic sensing for telecommunication satellites. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. *International Conference on Space Optics—ICSO 2008*. [S.l.], 2017. v. 10566, p. 105661C. Citado na página 16.
- RODRIGUEZ, J. E. O. *Processo de referência para o desenvolvimento da arquitetura de uma estação terrena para pico e nanosatélites*. Dissertação (Mestrado) — INPE, 2016. Acessado em: 20/01/2019. Disponível em: <<http://urlib.net/8JMKD3MGP3W34P/3LDAGLL>>. Citado na página 11.
- ROSCOSMOS. *Chronicle of Soviet-Russian space program*. 2016. Acessado em: 10/04/2019. Disponível em: <<http://en.federalspace.ru/174/>>. Citado na página 5.
- ROZENFELD, H. et al. *Gestão de desenvolvimento de produtos: uma referência para a melhoria do processo*. [S.l.: s.n.], 2006. Citado 4 vezes nas páginas 36, 37, 60 e 64.
- SCHUSTER, C. H.; SCHUSTER, J. J.; OLIVEIRA, A. S. de. Aplicação do diagrama de mudge e qfd utilizando como exemplo a hierarquização dos requisitos para um carro voador. *Revista GEPROS*, v. 10, n. 1, p. 197, 2015. Citado na página 63.
- SELVA, D.; KREJCI, D. A survey and assessment of the capabilities of cubesats for earth observation. *Acta Astronautica*, Elsevier, v. 74, p. 50–68, 2012. Citado 2 vezes nas páginas 8 e 14.
- STELLA, G. N. D. et al. *Aplicando a metodologia de desenvolvimento baseado em modelos para funções de software automotivo*. Dissertação (Mestrado) — Universidade Tecnológica Federal do Paraná, 2015. Citado 3 vezes nas páginas 50, 51 e 54.
- TANENBAUM, A. S. *Redes de Computadores*. 4^a. [S.l.: s.n.], 2003. Citado 2 vezes nas páginas 28 e 29.
- TECHNOLOGY, O. A. *Ramp-Up de Model-Based Design*. 2018. <https://opencadduniversity.com.br/home/lesson/ramp-up-de-model-based-design/2/50>. Acessado em: 12/05/2019. Citado na página 53.
- TOORIAN ARMEN; DIAZ, K. L. S. The cubesat approach to space access. *2008 IEEE Aerospace Conference*, p. 1–14, 2008. Citado 3 vezes nas páginas 1, 14 e 36.
- TP-LINK. *2.4GHz 5dBi Indoor Omnidirectional Antenna TL-ANT2405C*. 2017. Acessado em: 01/06/2019. Disponível em: <<https://www.tp-link.com/br/support/download/tl-ant2405c/>>. Citado na página 101.

TPU. *TPU Launched First Russia's 3D-Satellite*. 2016. Acessado em: 15/04/2019. Disponível em: <<https://ssel.montana.edu/printsat.html>>. Citado na página 9.

TWIGGS, R. Origin of cubesat. *Small Satellites: Past, Present, and Future*, Eds: Helvajian H., Janson SW, The Aerospace Press, El Segundo, California, 2008. Citado na página 6.

WOELLERT, K. et al. Cubesats: Cost-effective science and technology platforms for emerging and developing nations. *Advances in Space Research*, Elsevier, v. 47, n. 4, p. 663–684, 2011. Citado na página 10.

WU, J.-C.; LEU, M.; LIU, X. F. A hierarchical object-oriented functional modeling framework for co-design of mechatronic products. *Concurrent Engineering: R&A*, v. 17, p. 245–256, 12 2009. Citado 5 vezes nas páginas 2, 37, 38, 39 e 40.

ZHENG, C. et al. Survey on mechatronic engineering: A focus on design methods and product models. *Advanced Engineering Informatics*, v. 28, 08 2014. Citado na página 36.

ZHENG, C. et al. Multidisciplinary interface model for design of mechatronic systems. *Computers in Industry*, v. 76, p. 24–37, 02 2016. Citado 4 vezes nas páginas 2, 37, 38 e 42.

Apêndices

APÊNDICE A – Diagrama de Mudge

LEGENDA		
Simbolo	Critérios de comparação	Valor
A	Muito importante	5
B	Importante	3
C	Pouco importante	1

Requisitos do cliente	MUDGE																								
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
1 Transmissão de dados																									
2 Recepção de dados																									
3 Confirmação de entrega de dados																									
4 Coleta de dados																									
5 Baixo peso																									
6 Ocupar baixo volume																									
7 Faixa de frequência regulamentada																									
8 Potência de transmissão regulamentada																									
9 Possuir conectividade																									
10 Encapsular dados																									
11 Compatível com outros componentes																									
12 Tratar falhas de comunicação																									
13 Interface Homem-Máquina (IHM)																									
14 Armazenamento de dados																									
15 Upgrade de componentes																									
16 Operação em tempo real																									
17 Especificações técnicas bem definidas																									
18 Direcionalidade																									
19 Utilização de fontes de energia alternativas																									
20 Baixo custo																									
21 Adaptável a variações externas																									
22 Alta controlabilidade																									
23 Alta confiabilidade na operação																									
24 Alta qualidade																									
	A	0	0	3	1	1	5	5	0	0	0	0	0	0	0	0	1	0	1	9	6	10	10	11	
	B	0	0	0	3	0	0	0	1	0	3	0	3	0	1	1	3	1	3	1	10	2	1	0	3
	C	0	0	0	0	0	0	0	1	0	1	0	5	0	4	0	1	0	1	2	5	3	1	3	
	TOTAL COLUNAS	0	0	15	14	14	25	25	0	4	0	10	0	14	0	12	6	15	48	61	58	58	53	67	
	A	0	0	3	1	1	5	5	0	0	0	0	0	0	0	1	0	1	9	6	10	10	11		
	B	0	0	0	3	0	0	0	1	0	3	0	3	0	1	1	3	1	3	1	10	2	1	0	3
	C	0	0	0	0	0	0	0	1	0	1	0	5	0	4	0	1	0	1	2	5	3	1	3	
	TOTAL LINHAS	0	0	3	1	1	5	5	0	0	0	0	0	0	0	1	0	1	9	6	10	10	11		
	A	0	0	3	1	1	5	5	0	0	0	0	0	0	0	1	0	1	9	6	10	10	11		
	B	0	0	0	3	0	0	0	1	0	3	0	3	0	1	1	3	1	3	1	10	2	1	0	3
	C	0	0	0	0	0	0	0	1	0	1	0	5	0	4	0	1	0	1	2	5	3	1	3	
	TOTAL GERAL	0	0	15	14	14	25	25	0	4	0	10	0	14	0	12	6	15	48	61	58	58	53	67	

APÊNDICE B – Teste SIL - Código C

```

/*
 * Arquivo: ModeloVirtual.c
 *
 * Código gerado pelo modelo Simulink 'ModeloVirtual'.
 *
 * Versão do Modelo           : 1.13
 * Simulink Versão           : 8.10 (R2016a) 10-Feb-2016
 * C/C++ Código fonte Gerado em : 10 de Julho 14:50:30 2019
 *
 * Seleção de alvo: ert.tlc
 * Seleção de hardware embarcado: Atmel->AVR (8-bit)
 * Objetivos da Geração do Código:
 *   1. Eficiência na execução
 *   2. Eficiência RAM
 */

#include "ModeloVirtual.h"
#define NumBitsPerChar          8U

/* Macros particulares usadas pelo código gerado para acessar
   rtModel */
#ifndef rtmIsMajorTimeStep
# define rtmIsMajorTimeStep(rtm)      (((rtm)->Timing.simTimeStep) ==
MAJOR_TIME_STEP)
#endif

#ifndef rtmIsMinorTimeStep
# define rtmIsMinorTimeStep(rtm)     (((rtm)->Timing.simTimeStep) ==
MINOR_TIME_STEP)
#endif

#ifndef rtmGetTPtr
# define rtmGetTPtr(rtm)              ((rtm)->Timing.t)
#endif

#ifndef rtmSetTPtr
# define rtmSetTPtr(rtm, val)         ((rtm)->Timing.t = (val))
#endif

#include "dsp_rt.h" /* DSP Funções gerais de suporte ao
tempo de execução do System Toolbox */

/* Usado pelo bloco de espaço de trabalho: '<S16>/From Workspace' */
#ifndef rtInterpolate
# define rtInterpolate(v1,v2,f1,f2)   (((v1)==(v2))?(double)(v1):
((f1)*((double)(v1)))+(f2)*((double)(v2))))
#endif

#ifndef rtRound
# define rtRound(v)                   ( ((v) >= 0) ? floor((v) + 0.5) :
ceil((v) - 0.5) )
#endif

const creal_T ModeloVirtual_RGND_Complex = { 0.0, 0.0 }; /* real_T */

/* Bloquear sinais e estados (armazenamento automático) */
DW rtDW;

```

```

/* Modelo em Tempo Real */
RT_MODEL rtM_;
RT_MODEL *const rtM = &rtM_;
extern real_T rt_hypotd_snf(real_T u0, real_T u1);
extern real_T rt_atan2d_snf(real_T u0, real_T u1);
extern void MWDSPCG_R2BRScramble_O_2fCUGA4L(creal_T y[], const creal_T x[],
int32_T nChans, int32_T nRows);
extern void MWDSPCG_R2DIT_TBLS_Z(creal_T y[], int32_T nChans, int32_T nRows,
int32_T fftLen, int32_T offset, const real_T tablePtr[], int32_T
twiddleStep,
boolean_T isInverse);
extern void RandSrcInitState_GZ(const uint32_T seed[], uint32_T state[],
int32_T
nChans);
extern void RandSrc_GZ_Z(creal_T y[], const creal_T mean[], int32_T meanLen,
const real_T xstd[], int32_T xstdLen, uint32_T state[], int32_T nChans,
int32_T nSamps);
extern real_T rt_powd_snf(real_T u0, real_T u1);
void Look2DEvenEven_Cplx_re_JaThNA4p(real_T *pY, const real_T *pYData,
real_T u0,
real_T u0Lo, uint32_T idxU0Max, real_T u0Spacing, real_T u1, real_T u1Lo,
uint32_T idxU1Max, real_T u1Spacing);
void LookUpEven_real_T_real_T(real_T *pY, const real_T *pYData, real_T u,
real_T
valueLo, uint16_T iHi, real_T uSpacing);
extern void p2d(creal_T rtu_In1, creal_T *rty_Out1, ConstB_p2d *localC,
DW_p2d
*localDW);
extern void phasenoise_Init(DW_phasenoise *localDW);
extern void phasenoise(creal_T rtu_In, creal_T *rty_Out, DW_phasenoise
*localDW);
extern real_T rtGetInf(void);
extern real32_T rtGetInfF(void);
extern real_T rtGetMinusInf(void);
extern real32_T rtGetMinusInfF(void);
extern real_T rtGetNaN(void);
extern real32_T rtGetNaNF(void);

/*=====
 * Constantees *
 *=====*/
#define RT_PI 3.14159265358979323846
#define RT_PIF 3.1415927F
#define RT_LN_10 2.30258509299404568402
#define RT_LN_10F 2.3025851F
#define RT_LOG10E 0.43429448190325182765
#define RT_LOG10EF 0.43429449F
#define RT_E 2.7182818284590452354
#define RT_EF 2.7182817F

/*
 * UNUSED_PARAMETER(x)
 * Usado para especificar que um parâmetro de função (argumento) é
obrigatório, mas não
 * acessado pelo corpo da função.
 */

```

```

#ifndef UNUSED_PARAMETER
# if defined(__LCC__)
#   define UNUSED_PARAMETER(x)
# else

/*
 * Esta é a maneira padrão semi-ANSI de indicar que um
 * O parâmetro de função não utilizado é obrigatório.
 */
#   define UNUSED_PARAMETER(x)          (void) (x)
# endif
#endif

#ifndef INTERP
# define INTERP(x,x1,x2,y1,y2)          ( (y1)+(((y2) - (y1))/((x2) -
(x1))) * ((x)-(x1)) )
#endif

#ifndef ZEROTECHNIQUE
#define ZEROTECHNIQUE

typedef enum {
    NORMAL_INTERP,
    AVERAGE_VALUE,
    MIDDLE_VALUE
} ZeroTechnique;

#endif

extern int_T rt_GetLookupIndex(const real_T *x, int_T xlen, real_T u) ;
extern real_T rt_Lookup(const real_T *x, int_T xlen, real_T u, const real_T
*y);
extern real_T rtInf;
extern real_T rtMinusInf;
extern real_T rtNaN;
extern real32_T rtInfF;
extern real32_T rtMinusInfF;
extern real32_T rtNaNF;
extern void rt_InitInfAndNaN(size_t realSize);
extern boolean_T rtIsInf(real_T value);
extern boolean_T rtIsInfF(real32_T value);
extern boolean_T rtIsNaN(real_T value);
extern boolean_T rtIsNaNF(real32_T value);
typedef struct {
    struct {
        uint32_T wordH;
        uint32_T wordL;
    } words;
} BigEndianIEEEEDouble;

typedef struct {
    struct {
        uint32_T wordL;
        uint32_T wordH;
    } words;
} LittleEndianIEEEEDouble;

```

```

typedef struct {
    union {
        real32_T wordLreal;
        uint32_T wordLuint;
    } wordL;
} IEEESingle;

real_T rtInf;
real_T rtMinusInf;
real_T rtNaN;
real32_T rtInfF;
real32_T rtMinusInfF;
real32_T rtNaNF;

/*
 * Inicialize o rtInf necessário pelo código gerado.
 * Inf é inicializado como sem sinalização. AsSomae o IEEE.
 */
real_T rtGetInf(void)
{
    size_t bitsPerReal = sizeof(real_T) * (NumBitsPerChar);
    real_T inf = 0.0;
    if (bitsPerReal == 32U) {
        inf = rtGetInfF();
    } else {
        union {
            LittleEndianIEEEDouble bitVal;
            real_T fltVal;
        } tmpVal;

        tmpVal.bitVal.words.wordH = 0x7FF00000U;
        tmpVal.bitVal.words.wordL = 0x00000000U;
        inf = tmpVal.fltVal;
    }

    return inf;
}

/*
 * Inicialize o rtInfF necessário pelo código gerado.
 * Inf é inicializado como sem sinalização. AsSomae o IEEE.
 */
real32_T rtGetInfF(void)
{
    IEEESingle infF;
    infF.wordL.wordLuint = 0x7F800000U;
    return infF.wordL.wordLreal;
}

/*
 * Inicialize o rtMinusInf necessário pelo código gerado.
 * Inf é inicializado como sem sinalização. AsSomae o IEEE.
 */
real_T rtGetMinusInf(void)
{
    size_t bitsPerReal = sizeof(real_T) * (NumBitsPerChar);
    real_T minf = 0.0;

```

```

    if (bitsPerReal == 32U) {
        minf = rtGetMinusInfF();
    } else {
        union {
            LittleEndianIEEEEDouble bitVal;
            real_T fltVal;
        } tmpVal;

        tmpVal.bitVal.words.wordH = 0xFFF00000U;
        tmpVal.bitVal.words.wordL = 0x00000000U;
        minf = tmpVal.fltVal;
    }

    return minf;
}

/*
 * Inicialize o rtMinusInfF necessário pelo código gerado.
 * Inf é inicializado como sem sinalização. AsSomae o IEEE.
 */
real32_T rtGetMinusInfF(void)
{
    IEEEESingle minfF;
    minfF.wordL.wordLuint = 0xFF800000U;
    return minfF.wordL.wordLreal;
}

/*
 * Inicialize o rtNaN necessário pelo código gerado.
 * NaN é inicializado como sem sinalização. AsSomae o IEEE.
 */
real_T rtGetNaN(void)
{
    size_t bitsPerReal = sizeof(real_T) * (NumBitsPerChar);
    real_T nan = 0.0;
    if (bitsPerReal == 32U) {
        nan = rtGetNaNF();
    } else {
        union {
            LittleEndianIEEEEDouble bitVal;
            real_T fltVal;
        } tmpVal;

        tmpVal.bitVal.words.wordH = 0xFFF80000U;
        tmpVal.bitVal.words.wordL = 0x00000000U;
        nan = tmpVal.fltVal;
    }

    return nan;
}

/*
 * Inicialize o rtNaNF necessário pelo código gerado.
 * NaN é inicializado como sem sinalização. AsSomae o IEEE.
 */
real32_T rtGetNaNF(void)
{

```

```

IEEESingle nanF = { { 0 } };

nanF.wordL.wordLuint = 0xFFC00000U;
return nanF.wordL.wordLreal;
}

/*
 * Rotina para obter o índice da entrada de uma tabela usando binário ou
 * pesquisa de interpolação.
 *
 * Entradas:
 *   *x   : Tabela de ponteiros, x[0] ...x[xlen-1]
 *   xlen : numero de valores em xtable
 *   u    : valor de entrada
 *
 * Saídas:
 *   idx  : o índice na tabela de tal parama que:
 *           if u is negative
 *             x[idx] <= u < x[idx+1]
 *           else
 *             x[idx] < u <= x[idx+1]
 *
 * Pesquisa de interpolação: se a tabela contiver um grande número de
 * entradas uniparamemente espaçadas, isto é, x [n] vs n é linear, então o
 * índice correspondente à entrada pode ser encontrado em um tiro usando o
 * linear fórmula de interpolação. Portanto, se você tiver um bloco de consulta
 * com muitos pontos de dados, usando a pesquisa de interpolação, podem
 * acelerar o código. Compile o código gerado com o seguinte sinalizador:
 *
 *           make_rtw OPTS=-DDOINTERPSEARCH
 *
 * para ativar a pesquisa de interpolação.
 */
int_T rt_GetLookupIndex(const real_T *x, int_T xlen, real_T u)
{
    int_T idx = 0;
    int_T bottom = 0;
    int_T top = xlen-1;
    int_T retValue = 0;
    boolean_T returnStatus = 0U;

#ifdef DDOINTERPSEARCH

    real_T offset = 0;

#endif

    /*
     * Lide com os casos extremos primeiro:
     *   if u <= x[bottom] then return idx = bottom
     *   if u >= x[top]    then return idx = top-1
     */
    if (u <= x[bottom]) {
        retValue = bottom;
        returnStatus = 1U;
    } else if (u >= x[top]) {
        retValue = top-1;
    }
}

```



```

    returnStatus = 1U;
} else {
    /* necessário para garantir a programação segura, mesmo *
    * se é esperado que nunca seja alcançado */
}

if (returnStatus == 0U) {
    if (u < 0) {
        /* Para entrada negativa, encontre um índice tal que: x[idx] <= u <
x[idx+1] */
        para (;;) {

#ifdef DOINTERPSEARCH

            offset = (u-x[bottom])/(x[top]-x[bottom]);
            idx = bottom + (int_T)((top-bottom)*(offset-DBL_EPSILON));

#else

            idx = (bottom + top)/2;

#endif

            if (u < x[idx]) {
                top = idx - 1;
            } else if (u >= x[idx+1]) {
                bottom = idx + 1;
            } else {
                /* we have x[idx] <= u < x[idx+1], return idx */
                retValue = idx;
                break;
            }
        }
    } else {
        /* Para entrada não negativa, localize um índice tal que: x[idx] < u
<= x[idx+1] */
        para (;;) {

#ifdef DOINTERPSEARCH

            offset = (u-x[bottom])/(x[top]-x[bottom]);
            idx = bottom + (int_T)((top-bottom)*(offset-DBL_EPSILON));

#else

            idx = (bottom + top)/2;

#endif

            if (u <= x[idx]) {
                top = idx - 1;
            } else if (u > x[idx+1]) {
                bottom = idx + 1;
            } else {
                /* we have x[idx] < u <= x[idx+1], return idx */
                retValue = idx;
                break;
            }
        }
    }
}

```

```

    }
    }
}

return retValue;
}

/* 1D rotina de pesquisa para tipo de dados de tempo real. */
real_T rt_Lookup(const real_T *x, int_T xlen, real_T u, const real_T *y)
{
    int_T idx = rt_GetLookupIndex(x, xlen, u);
    real_T num = y[idx+1] - y[idx];
    real_T den = x[idx+1] - x[idx];

    /* Devido à maneira como a pesquisa binária é implementada
       em rt_look.c (rt_GetLookupIndex), den não pode ser
       0. Equivalente, m não pode ser inf ou nan. */
    real_T m = num/den;
    return (y[idx] + (m * (u - x[idx])));
}

/*
 * Inicialize o rtInf, rtMinusInf, and rtNaN necessário para
 * geração do código. NaN é inicializado como sem sinalização. AsSomae o IEEE.
 */
void rt_InitInfAndNaN(size_t realSize)
{
    (void) (realSize);
    rtNaN = rtGetNaN();
    rtNaNF = rtGetNaNF();
    rtInf = rtGetInf();
    rtInfF = rtGetInfF();
    rtMinusInf = rtGetMinusInf();
    rtMinusInfF = rtGetMinusInfF();
}

/* Teste se o valor é infinito */
boolean_T rtIsInf(real_T value)
{
    return (boolean_T)((value==rtInf || value==rtMinusInf) ? 1U : 0U);
}

/* Teste se o valor de precisão única para infinito */
boolean_T rtIsInfF(real32_T value)
{
    return (boolean_T)(((value)==rtInfF || (value)==rtMinusInfF) ? 1U : 0U);
}

/* Teste se o valor não para um número */
boolean_T rtIsNaN(real_T value)
{
    return (boolean_T)((value!=value) ? 1U : 0U);
}

/* Teste se o valor de precisão simples não para um número */
boolean_T rtIsNaNF(real32_T value)

```

```

{
    return (boolean_T)((value!=value) ? 1U : 0U));
}

/* Lookup 2D Utilitário de pesquisa Look2DEvenEven_Cplx_re_JaThNA4p */
void Look2DEvenEven_Cplx_re_JaThNA4p(real_T *pY, const real_T *pYData,
real_T u0,
    real_T u0Lo, uint32_T idxU0Max, real_T u0Spacing, real_T u1, real_T u1Lo,
    uint32_T idxU1Max, real_T u1Spacing)
{
    uint32_T idxU0Left = 0U;
    uint32_T idxU0Rght = 0U;
    uint32_T idxU1Left = 0U;
    uint32_T idxU1Rght = 0U;
    real_T u0MinusLeft = 0.0;
    real_T u1MinusLeft = 0.0;
    if (u0 <= u0Lo ) {
        u0MinusLeft = 0.0;
        idxU0Left = 0U;
        idxU0Rght = 0U;
    } else {
        real_T tmpIdxLeft;
        u0MinusLeft = (real_T)(u0 - u0Lo);
        tmpIdxLeft = u0MinusLeft / u0Spacing;
        idxU0Left = (uint32_T)tmpIdxLeft;
        if ((tmpIdxLeft >= 4294967296.0) || (idxU0Left >= idxU0Max)) {
            u0MinusLeft = 0.0;
            idxU0Left = idxU0Max;
            idxU0Rght = idxU0Max;
        } else {
            u0MinusLeft = u0MinusLeft - (idxU0Left * u0Spacing);
            idxU0Rght = idxU0Left + 1U;
        }
    }

    if (u1 <= u1Lo ) {
        u1MinusLeft = 0.0;
        idxU1Left = 0U;
        idxU1Rght = 0U;
    } else {
        real_T tmpIdxLeft;
        u1MinusLeft = (real_T)(u1 - u1Lo);
        tmpIdxLeft = u1MinusLeft / u1Spacing;
        idxU1Left = (uint32_T)tmpIdxLeft;
        if ((tmpIdxLeft >= 4294967296.0) || (idxU1Left >= idxU1Max)) {
            u1MinusLeft = 0.0;
            idxU1Left = idxU1Max;
            idxU1Rght = idxU1Max;
        } else {
            u1MinusLeft = u1MinusLeft - (idxU1Left * u1Spacing);
            idxU1Rght = idxU1Left + 1U;
        }
    }

    {
        real_T yTemp;
        real_T yLeftLeft;
    }
}

```

```

real_T yLeftRght;
real_T yRghtLeft;
real_T yRghtRght;
real_T u1Lambda;
real_T u0Lambda;
u1Lambda = u1MinusLeft / u1Spacing;
u0Lambda = u0MinusLeft / u0Spacing;
idxU0Max++;
idxU1Left *= idxU0Max;
idxU1Rght *= idxU0Max;
yRghtLeft = pYData[2*((idxU0Rght+idxU1Left))];
yRghtRght = pYData[2*((idxU0Rght+idxU1Rght))];
yLeftLeft = pYData[2*((idxU0Left+idxU1Left))];
yLeftRght = pYData[2*((idxU0Left+idxU1Rght))];

/* Interpolar pela variável u1
 * com a variável u0 bloqueada à esquerda u0
 */
{
    real_T yLeftCast;
    real_T yRghtCast;
    yLeftCast = yLeftLeft;
    yRghtCast = yLeftRght;
    yLeftCast += u1Lambda * ( yRghtCast - yLeftCast );
    pY[0] = yLeftCast;
}

/* Interpolar pela variável u1
 * com a variável u0 bloqueada à direita u0
 */
{
    real_T yLeftCast;
    real_T yRghtCast;
    yLeftCast = yRghtLeft;
    yRghtCast = yRghtRght;
    yLeftCast += u1Lambda * ( yRghtCast - yLeftCast );
    yTemp = yLeftCast;
}

/*
 * Interpolar pela variável u0
 * com a variável u1 bloqueada em seu valor interpolado
 */
{
    real_T yLeftCast;
    real_T yRghtCast;
    yLeftCast = pY[0];
    yRghtCast = yTemp;
    yLeftCast += u0Lambda * ( yRghtCast - yLeftCast );
    pY[0] = yLeftCast;
}

yRghtLeft = pYData[2*((idxU0Rght+idxU1Left))+1];
yRghtRght = pYData[2*((idxU0Rght+idxU1Rght))+1];
yLeftLeft = pYData[2*((idxU0Left+idxU1Left))+1];
yLeftRght = pYData[2*((idxU0Left+idxU1Rght))+1];

```

```

/* Interpolar pela variável u0
 *   com a variável u0 bloqueada à esquerda   */
{
    real_T yLeftCast;
    real_T yRghtCast;
    yLeftCast = yLeftLeft;
    yRghtCast = yLeftRght;
    yLeftCast += u0Lambda * ( yRghtCast - yLeftCast );
    pY[1] = yLeftCast;
}

/* Interpolar pela variável u1
 *   com a variável u0 bloqueada à direita   */
{
    real_T yLeftCast;
    real_T yRghtCast;
    yLeftCast = yRghtLeft;
    yRghtCast = yRghtRght;
    yLeftCast += u1Lambda * ( yRghtCast - yLeftCast );
    yTemp = yLeftCast;
}

/*
 * Interpolar pela variável u0
 *   com a variável u1 bloqueada em seu valor interpolado
 */
{
    real_T yLeftCast;
    real_T yRghtCast;
    yLeftCast = pY[1];
    yRghtCast = yTemp;
    yLeftCast += u0Lambda * ( yRghtCast - yLeftCast );
    pY[1] = yLeftCast;
}
}

}

/* Lookup 1D UtilityLookUpEven_real_T_real_T */
void LookUpEven_real_T_real_T(real_T *pY, const real_T *pYData, real_T u,
real_T
valueLo, uint16_T iHi, real_T uSpacing)
{
    if (u <= valueLo ) {
        (*pY) = (*pYData);
    } else {
        real_T uAdjusted = u - valueLo;
        real_T tmpIdxLeft = uAdjusted / uSpacing;
        uint32_T iLeft = (uint32_T)tmpIdxLeft;
        if ((tmpIdxLeft >= 4294967296.0) || (iLeft >= iHi) ) {
            (*pY) = pYData[iHi];
        } else {
            {
                real_T lambda;

                {
                    real_T num = (real_T)uAdjusted - ( iLeft * uSpacing );
                    lambda = num / uSpacing;

```

```

    }

    {
        real_T yLeftCast;
        real_T yRightCast;
        yLeftCast = pYData[iLeft];
        yRightCast = pYData[(iLeft)+1];
        yLeftCast += lambda * ( yRightCast - yLeftCast );
        (*pY) = yLeftCast;
    }
}
}
}
}

```

```

real_T rt_hypotd_snf(real_T u0, real_T u1)

```

```

{
    real_T y;
    real_T a;
    a = fabs(u0);
    y = fabs(u1);
    if (a < y) {
        a /= y;
        y *= sqrt(a * a + 1.0);
    } else if (a > y) {
        y /= a;
        y = sqrt(y * y + 1.0) * a;
    } else {
        if (!rtIsNaN(y)) {
            y = a * 1.4142135623730951;
        }
    }

    return y;
}

```

```

real_T rt_atan2d_snf(real_T u0, real_T u1)

```

```

{
    real_T y;
    int16_T u0_0;
    int16_T u1_0;
    if (rtIsNaN(u0) || rtIsNaN(u1)) {
        y = (rtNaN);
    } else if (rtIsInf(u0) && rtIsInf(u1)) {
        if (u0 > 0.0) {
            u0_0 = 1;
        } else {
            u0_0 = -1;
        }

        if (u1 > 0.0) {
            u1_0 = 1;
        } else {
            u1_0 = -1;
        }
    }

    y = atan2(u0_0, u1_0);
}

```

```

} else if (u1 == 0.0) {
    if (u0 > 0.0) {
        y = RT_PI / 2.0;
    } else if (u0 < 0.0) {
        y = -(RT_PI / 2.0);
    } else {
        y = 0.0;
    }
} else {
    y = atan2(u0, u1);
}

return y;
}

void MWDSPCG_R2BRScramble_O_2fCUGA4L(creal_T y[], const creal_T x[], int32_T
nChans, int32_T nRows)
{
    int32_T yIdx;
    int32_T j;
    int32_T i;
    int32_T bit_fftLen;

    /* S-Function (sdspfft2): '<S48>/IFFT' */
    /* algoritmo paraa do lugar */
    yIdx = 0L;
    while (nChans
        > 0L) {
        nChans
            --;
        j = 0L;

        /* Para cada element no array de origem */
        para (i = 0L; i
            < nRows
            - 1L; i
            ++) {
            /* Copiar element na posição de bit-rev */
            y[j
                + yIdx] = x[i
                    + yIdx];

            /* Calcular o próximo índice de destino com reversão de bits
            */
            bit_fftLen = nRows;
            do {
                bit_fftLen = (int32_T)((uint32_T)bit_fftLen
                    >> 1);
                j
                    ^= bit_fftLen;
            } while (
                !((j
                    & bit_fftLen)
                    != 0L));
        }

        /* Copiar elemento final */

```

```

    y[j
      + yIdx] = x[i
      + yIdx];
    yIdx
      += nRows;
  }

  /* Fim do S-Function (sdspfft2): '<S48>/IFFT' */
}

void MWDSPCG_R2DIT_TBLS_Z(creal_T y[], int32_T nChans, int32_T nRows, int32_T
  fftLen, int32_T offset, const real_T tablePtr[], int32_T twiddleStep,
  boolean_T isInverse)
{
  int32_T nHalf;
  real_T twidRe;
  real_T twidIm;
  int32_T nQtr;
  int16_T fwdInvFactor;
  int32_T iCh;
  int32_T offsetCh;
  int32_T idelta;
  int32_T ix;
  int32_T k;
  int32_T kratio;
  int32_T istart;
  int32_T il;
  int32_T j;
  int32_T i2;
  real_T tmp_re;
  real_T tmp_im;

  /* S-Function (sdspfft2): '<S48>/IFFT' */
  /* DSP System Toolbox Decimation in Time FFT */
  /* Computação realizada usando pesquisa de tabela
  */
  /* Tipo de saída: complexa real_T */
  nHalf = (fftLen
    >> 1)
    * twiddleStep;
  nQtr = nHalf
    >> 1;
  fwdInvFactor = isInverse
    ? -1
    : 1;

  /* Para cada canal */
  offsetCh = offset;
  para (iCh = 0L; iCh
    < nChans; iCh
    ++) {
    /* Execute condições para o primeiro estágio, onde não é necessário
    multiplicar
    . */
    para (ix = offsetCh; ix
      < (fftLen
        + offsetCh)

```



```

        - 1L; ix
        += 2L) {
    i2 = ix
        + 1L;
    tmp_re = y[i2].re;
    tmp_im = y[i2].im;
    y[i2].re = y[ix].re
        - tmp_re;
    y[i2].im = y[ix].im
        - tmp_im;
    y[ix].re
        += tmp_re;
    y[ix].im
        += tmp_im;
}

idelta = 2L;
k = fftLen
    >> 2;
kratio = k
    * twiddleStep;
while (k
        > 0L) {
    i1 = offsetCh;

    /* Realize a primeira condição em cada etapa restante, onde
    nenhuma multiplicação é necessária.
    */
    para (ix = 0L; ix
        < k; ix
        ++) {
        i2 = i1
            + idelta;
        tmp_re = y[i2].re;
        tmp_im = y[i2].im;
        y[i2].re = y[i1].re
            - tmp_re;
        y[i2].im = y[i1].im
            - tmp_im;
        y[i1].re
            += tmp_re;
        y[i1].im
            += tmp_im;
        i1
            += idelta
            << 1;
    }

    istart = offsetCh;

    /* Execute as condições restantes */
    para (j = kratio; j
        < nHalf; j
        += kratio) {
        i1 = istart
            + 1L;
        twidRe = tablePtr[j];

```

```

    twidIm = tablePtr[j
    + nQtr]
    * (real_T) fwdInvFactor;
para (ix = 0L; ix
    < k; ix
    ++ ) {
    i2 = i1
    + idelta;
    tmp_re = y[i2].re
    * twidRe
    - y[i2].im
    * twidIm;
    tmp_im = y[i2].re
    * twidIm
    + y[i2].im
    * twidRe;
    y[i2].re = y[i1].re
    - tmp_re;
    y[i2].im = y[i1].im
    - tmp_im;
    y[i1].re
    += tmp_re;
    y[i1].im
    += tmp_im;
    i1
    += idelta
    << 1;
    }

    istart
    ++;
}

idelta
<<= 1;
k
>>= 1;
kratio
>>= 1;
}

/* Aponte para o próximo canal */
offsetCh
+= nRows;
}

/* Fim do S-Function (sdspfft2): '<S48>/IFFT' */
}

/*
* Saída e atualização para o sistema de ação
:
* '<S28>/p2d'
* '<S30>/p2d'
* '<S91>/p2d'
* '<S93>/p2d'
*/

```

```

void p2d(creal_T rtu_In1, creal_T *rty_Out1, ConstB_p2d *localC, DW_p2d
*localDW)
{
    int32_T s46_iter;
    int16_T colIdx;
    real_T rtb_Saturation;
    real_T rtb_ComplextomagnitudeAngle_o2;
    creal_T rtb_IFFT[2];
    real_T rtb_Add_idx_1;

    /* Saídas para o subsistema de iteração: '<S36>/Para Iterator Subsystem1'
    incorporados:
    * ParaIterator: '<S46>/Para Iterator'
    */
    para (s46_iter = 1L; s46_iter
        <= (int32_T)localC->Width1; s46_iter
        ++) {
        /* Ganho: '<S46>/Ganho' incorporados:
        * Constante: '<S46>/Constante1'
        * Soma: '<S46>/Add1'
        */
        rtb_Saturation = ((real_T)s46_iter
            - 1.0)
            * 2.0;

        /* Soma: '<S46>/Add' */
        rtb_ComplextomagnitudeAngle_o2 = rtb_Saturation
            + 1.0;
        rtb_Add_idx_1 = rtb_Saturation
            + 2.0;

        /* ComplexToMagnitudeAngle: '<S48>/Abs' incorporados:
        * Seleção: '<S46>/Selector'
        */
        rtb_Saturation = rt_hypotd_snf(rtu_In1.re, rtu_In1.im);

        /* Lookup2D: '<S48>/Lookup Table (2-D)'
        * Sobre '<S48>/Lookup Table (2-D)':
        * Input0 Real Tipo de dado: Ponto Flutuante real_T
        * Input1 Real Tipo de dado: Ponto Flutuante real_T
        * Output0 Complex Tipo de dado: Ponto Flutuante real_T
        * Metodo Lookup: Linear_Endpoint
        *
        * Row Data parâmetro usa o mesmo tipo de dados e escala como
        Input0
        * O parâmetro Column Data usa o mesmo tipo de dados e escala
        Input1
        * O parâmetro Table Data usa o mesmo tipo de dados e escala
        Output0
        */
        Look2DEvenEven_Cplx_re_JaThNA4p( (real_T
        *)(&(localDW->LookupTable2D[0])),
            (real_T *) (rtConstP.pooled18), rtb_Saturation, 0.0, 1U, 1.0, (1.0),
            1.0,
            1U, 1.0);
        Look2DEvenEven_Cplx_re_JaThNA4p( (real_T
        *)(&(localDW->LookupTable2D[1])),

```

```

    (real_T *) (rtConstP.pooledl8), rtb_Saturation, 0.0, 1U, 1.0, (2.0),
1.0,
    1U, 1.0);

/* S-Function (sdspsubmtrx): '<S48>/Submatrix' */
colIdx = 0;
while (colIdx
    <= 0) {
    localDW->fftshift[0] = localDW->LookupTable2D[1L];
    colIdx = 1;
}

/* Fim do S-Function (sdspsubmtrx): '<S48>/Submatrix' */

/* S-Function (sdspsubmtrx): '<S48>/Submatrix1' */
colIdx = 0;
while (colIdx
    <= 0) {
    localDW->fftshift[1] = localDW->LookupTable2D[0L];
    colIdx = 1;
}

/* Fim do S-Function (sdspsubmtrx): '<S48>/Submatrix1' */

/* S-Function (sdspfft2): '<S48>/IFFT' */
MWDSPCG_R2BRScramble_0_2fcUGA4L(&rtb_IFFT[0UL],
&localDW->fftshift[0UL], 1L,
    2L);
MWDSPCG_R2DIT_TBLS_Z(&rtb_IFFT[0UL], 1L, 2L, 2L, 0L, &rtConstP.pooledl1,
1L,
    true);

/* Transparamação inversa de escala */
rtb_IFFT[0L].re /= 2.0;
rtb_IFFT[0L].im /= 2.0;
rtb_IFFT[1L].re /= 2.0;
rtb_IFFT[1L].im /= 2.0;

/* Tarefa: '<S46>/Assignment' */
if (s46_iter
    == 1L) {
    localDW->Assignment[0] = localDW->Reshape[0];
    localDW->Assignment[1] = localDW->Reshape[1];
}

localDW->Assignment[(int16_T)rtb_ComplextomagnitudeAngle_o2
    - 1] = rtb_IFFT[0];
localDW->Assignment[(int16_T)rtb_Add_idx_1
    - 1] = rtb_IFFT[1];

/* Fim da Tarefa: '<S46>/Assignment' */
}

/* Fim das saídas para o subsistema
: '<S36>/Para Interação Subsystem1' */

/* DiscreteFir: '<S36>/Discrete FIR Filter' */

```

```

/* ConSomair linha de atraso e inicio de amostras de entrada
*/
rtb_ComplextoMagnitudeAngle_o2 = 0.0;
rtb_Add_idx_1 = 0.0;
s46_iter = 0L;
while ((int16_T)s46_iter
      < 1) {
    rtb_ComplextoMagnitudeAngle_o2
    += rtu_In1.re
    * localDW->Assignment[0L].re
    - rtu_In1.im
    * localDW->Assignment[0L].im;
    rtb_Add_idx_1
    += rtu_In1.re
    * localDW->Assignment[0L].im
    + rtu_In1.im
    * localDW->Assignment[0L].re;
    s46_iter = 1L;
}

while (s46_iter
      - 1L
      < 1L) {
    rtb_ComplextoMagnitudeAngle_o2
    += localDW->DiscreteFIRFilter_states.re
    * localDW->Assignment[s46_iter].re
    - localDW->DiscreteFIRFilter_states.im
    * localDW->Assignment[s46_iter].im;
    rtb_Add_idx_1
    += localDW->DiscreteFIRFilter_states.re
    * localDW->Assignment[s46_iter].im
    + localDW->DiscreteFIRFilter_states.im
    * localDW->Assignment[s46_iter].re;
    s46_iter
    ++;
}

/* Atualizar linha de atraso para o próximo quadro
*/
localDW->DiscreteFIRFilter_states = rtu_In1;

/* ComplexToMagnitudeAngle: '<S47>/Complex to Magnitude-Angle'
incorporados:
* DiscreteFir: '<S36>/Discrete FIR Filter'
*/
rtb_Saturation = rt_hypotd_snf(rtb_ComplextoMagnitudeAngle_o2,
rtb_Add_idx_1);
rtb_ComplextoMagnitudeAngle_o2 = rt_atan2d_snf(rtb_Add_idx_1,
rtb_ComplextoMagnitudeAngle_o2);

/* Saturaçãõ: '<S47>/Saturation' */
if (rtb_Saturation
    <= 0.0) {
    rtb_Saturation = 0.0;
}

/* Fim da Saturaçãõ: '<S47>/Saturation' */

```

```

/* MagnitudeAngleToComplex: '<S47>/Magnitude-Angle to Complex' */
rty_Out1->re = rtb_Saturation
* cos(rtb_ComplextMagnitudeAngle_o2);
rty_Out1->im = rtb_Saturation
* sin(rtb_ComplextMagnitudeAngle_o2);
}

void RandSrcInitState_GZ(const uint32_T seed[], uint32_T state[], int32_T
nChans)
{
    int32_T i;

    /* Condições de Inicialização para S-Function (sdsprandsrc2): '<S61>/Random
Source' */
    /* RandSrcInitState_GZ */
    para (i = 0L; i
        < nChans; i
            ++) {
        state[i
            << 1] = 362436069UL;
        state[(i
            << 1)
            + 1L] = seed[i]
            == 0UL
            ? 521288629UL
            : seed[i];
    }

    /* Fim da Condições de Inicialização para S-Function (sdsprandsrc2):
'<S61>/Random Source' */
}

void RandSrc_GZ_Z(creal_T y[], const creal_T mean[], int32_T meanLen, const
real_T xstd[], int32_T xstdLen, uint32_T state[], int32_T
nChans, int32_T nSamps)
{
    int32_T i;
    int32_T j;
    real_T r;
    real_T x;
    real_T s;
    real_T y_0;
    int32_T chan;
    real_T std;
    uint32_T icng;
    uint32_T jsr;
    int32_T samp;
    real_T resultsVal[2];
    real_T mean_0[2];
    static const real_T vt[65] = { 0.340945, 0.4573146, 0.5397793, 0.6062427,
        0.6631691, 0.7136975, 0.7596125, 0.8020356, 0.8417227, 0.8792102,
0.9148948,
        0.9490791, 0.9820005, 1.0138492, 1.044781, 1.0749254, 1.1043917,
1.1332738,
        1.161653, 1.189601, 1.2171815, 1.2444516, 1.2714635, 1.298265,
1.3249008,

```

```

1.3514125, 1.3778399, 1.4042211, 1.4305929, 1.4569915, 1.4834527,
1.5100122,
1.5367061, 1.5635712, 1.5906454, 1.617968, 1.6455802, 1.6735255,
1.7018503,
1.7306045, 1.7598422, 1.7896223, 1.8200099, 1.851077, 1.8829044,
1.9155831,
1.9492166, 1.9839239, 2.0198431, 2.0571356, 2.095993, 2.136645,
2.1793713,
2.2245175, 2.2725186, 2.3239338, 2.3795008, 2.4402218, 2.5075117,
2.5834658,
2.6713916, 2.7769942, 2.7769942, 2.7769942, 2.7769942 };

```

```

/* S-Function (sdsprandsrc2): '<S61>/Random Source' */
/* RandSrc_GZ_Z */

```

```

nSamps
+= nSamps;
para (chan = 0L; chan
      < nChans; chan
      ++) {
  std = xstd[xstdLen
            > 1L
            ? chan
            : 0L];
  icng = state[chan
               << 1];
  jsr = state[(chan
               << 1)
              + 1L];
  mean_0[0UL] = mean[meanLen
                    > 1L
                    ? chan
                    : 0L].re;
  mean_0[1UL] = mean[meanLen
                    > 1L
                    ? chan
                    : 0L].im;
para (samp = 0L; samp
      < nSamps; samp
      ++) {
  icng = 69069UL
        * icng
        + 1234567UL;
  jsr
    ^= jsr
    << 13L;
  jsr
    ^= jsr
    >> 17;
  jsr
    ^= jsr
    << 5L;
  i = (int32_T)(icng
               + jsr);
  j = (i
       & 63L)
    + 1L;
  r = (real_T)i

```

```

* 4.6566128730773926E-10
* vt[j];
if (!(fabs(r)
    <= vt[j]
    - 1L)) {
x = (fabs(r)
    - vt[j]
    - 1L)
    / (vt[j]
    - vt[j]
    - 1L);
icng = 69069UL
    * icng
    + 1234567UL;
jsr
    ^= jsr
    << 13L;
jsr
    ^= jsr
    >> 17;
jsr
    ^= jsr
    << 5L;
y_0 = (real_T)(int32_T)(icng
    + jsr)
    * 2.328306436538696E-10
    + 0.5;
s = x
    + y_0;
if (s
    > 1.301198) {
r = r
    < 0.0
    ? 0.4878992
    * x
    - 0.4878992
    : 0.4878992
    - 0.4878992
    * x;
} else {
if (!(s
    <= 0.9689279)) {
x = 0.4878992
    - 0.4878992
    * x;
if (y_0
    > 12.67706
    - exp(-0.5
        * x
        * x)
    * 12.37586) {
r = r
    < 0.0
    ?
    -x
    : x;
} else {

```



```

    }

    resultsVal[(int32_T)((int16_T)samp
                        & 1)] = mean_0[(int32_T)((int16_T)samp
                        + 1)]
    + std
    * r;
    if (((int16_T)samp
        & 1)
        != 0) {
        y[chan
          * (nSamps
            >> 1)
          + (samp
            >> 1)].re = resultsVal[0UL];
        y[chan
          * (nSamps
            >> 1)
          + (samp
            >> 1)].im = resultsVal[1UL];
    }
}

state[chan
  << 1] = icng;
state[(chan
  << 1)
  + 1L] = jsr;
}

/* Fim da S-Function (sdsprandsrc2): '<S61>/Fonte Aleatória */
}

/*
 * Sistema inicializado para o Sistema de ação:
 * '<S53>/phasenoise'
 * '<S96>/phasenoise'
 * '<S116>/phasenoise'
 */
void phasenoise_Init(DW_phasenoise *localDW)
{
    /* Condições de Inicialização para S-Function (sdsprandsrc2): '<S61>/ Fonte
    Aleatória' */
    localDW->RandomSource_SEED_DWORK = 1UL;
    RandSrcInitState_GZ(&localDW->RandomSource_SEED_DWORK,
                        localDW->RandomSource_STATE_DWORK, 1L);
}

/*
 * Saída e atualização para o sistema de ação:
 * '<S53>/phasenoise'
 * '<S96>/phasenoise'
 * '<S116>/phasenoise'
 */
void phasenoise(creal_T rtu_In, creal_T *rty_Out, DW_phasenoise *localDW)
{
    /* variáveis locais do bloco i/o */

```

```

creal_T rtb_Inherit_l;
creal_T rtb_MagnitudeAngletoComplex_n;

/* S-Function (sdsprandsrc2): '<S61>/Fonte Aleatória' */
RandSrc_GZ_Z(&rtb_Inherit_l, &rtConstP.pooled16, 1L, &rtConstP.pooled2,
1L,
            localDW->RandomSource_STATE_DWORK, 1L, 1L);

/* Produto: '<S61>/Produto' */
rtb_MagnitudeAngletoComplex_n.re = rtb_Inherit_l.re
* 1.4142135623730951;
rtb_MagnitudeAngletoComplex_n.im = rtb_Inherit_l.im
* 1.4142135623730951;

/* DSP System Toolbox Conversão de Status do Quadro (sdspfrmconv) -
<S62>/Herdar */
rtb_Inherit_l = rtb_MagnitudeAngletoComplex_n;

/* ComplexToRealImag: '<S60>/Complexo para Real-Imag' */
localDW->ComplextoRealImag = rtb_Inherit_l.re;

/* MagnitudeAngleToComplex: '<S60>/Magnitude-Angle to Complex'
incorporados:
* DiscreteFir: '<S60>/Filtro Discreto Discrete'
*/
rtb_MagnitudeAngletoComplex_n.re = cos(localDW->ComplextoRealImag);
rtb_MagnitudeAngletoComplex_n.im = sin(localDW->ComplextoRealImag);

/* Produto: '<S60>/Produto' */
rty_Out->re = rtu_In.re
* rtb_MagnitudeAngletoComplex_n.re
- rtu_In.im
* rtb_MagnitudeAngletoComplex_n.im;
rty_Out->im = rtu_In.re
* rtb_MagnitudeAngletoComplex_n.im
+ rtu_In.im
* rtb_MagnitudeAngletoComplex_n.re;
}

real_T rt_powd_snf(real_T u0, real_T u1)
{
    real_T y;
    real_T tmp;
    real_T tmp_0;
    if (rtIsNaN(u0) || rtIsNaN(u1)) {
        y = (rtNaN);
    } else {
        tmp = fabs(u0);
        tmp_0 = fabs(u1);
        if (rtIsInf(u1)) {
            if (tmp == 1.0) {
                y = (rtNaN);
            } else if (tmp > 1.0) {
                if (u1 > 0.0) {
                    y = (rtInf);
                } else {
                    y = 0.0;
                }
            }
        }
    }
}

```

```

    }
    } else if (u1 > 0.0) {
        y = 0.0;
    } else {
        y = (rtInf);
    }
} else if (tmp_0 == 0.0) {
    y = 1.0;
} else if (tmp_0 == 1.0) {
    if (u1 > 0.0) {
        y = u0;
    } else {
        y = 1.0 / u0;
    }
} else if (u1 == 2.0) {
    y = u0 * u0;
} else if ((u1 == 0.5) && (u0 >= 0.0)) {
    y = sqrt(u0);
} else if ((u0 < 0.0) && (u1 > floor(u1))) {
    y = (rtNaN);
} else {
    y = pow(u0, u1);
}
}

return y;
}

/* Função de passo do modelo TIDO */
void ModeloVirtual_step0(void) /* Sample time: [0.0s, 0.0s] */
{
    /* variáveis locais do bloco i/o */
    creal_T rtb_Inherit;
    creal_T rtb_DopplerFrequencyShift;
    real_T rtb_LookUpTable_k;
    real_T rtb_LookUpTable_j;
    real_T rtb_LookUpTable_a;
    real_T rtb_LookUpTable_f;
    real_T rtb_LookUpTable_b0;
    real_T rtb_LookUpTable_gc;
    real_T rtb_LookUpTable_o;
    real_T rtb_LookUpTable_kw;
    real_T rtb_RadarSignalDegradation;
    int16_T j;
    real_T rtb_ComplextorealImag_o2;
    real_T rtb_ComplextomagnitudeAngle1__j;
    creal_T rtb_SineWave;
    creal_T Rffilt;
    creal_T Rffilt_f;
    creal_T Rffilt_i;
    creal_T Rffilt_fz;
    real_T rtb_ComplextomagnitudeAngle1_n1;
    real_T rtb_SomaofElements_re;
    real_T rtb_SomaofElements_im;
    int32_T k;

    /* DigitalClock: '<S1>/Digital Clock' */

```

```

rtb_RadarSignalDegradation =
((rtM->Timing.clockTick1+rtM->Timing.clockTickH1*
  4294967296.0)) * 1.5625E-5);

/* Soma: '<S1>/Soma' incorporados:
 * Constantee: '<S1>/Constante'
 * Ganho: '<S1>/Ganho'
 */
rtb_RadarSignalDegradation = 30.0
 * rtb_RadarSignalDegradation
 + 10000.0;

/* DiscretePulseGenerator: '<S5>/Gerador de Pulso' */
rtb_ComplectoRealImag_o2 = (rtDW.clockTickCounter
 < 128L)
 && (rtDW.clockTickCounter
 >= 0L));
if (rtDW.clockTickCounter
 >= 6399L) {
  rtDW.clockTickCounter = 0L;
} else {
  rtDW.clockTickCounter
  ++;
}

/* Fim do DiscretePulseGenerator: '<S5>/Gerador de Pulso' */

/* FromWorkspace: '<S16>/Do Espaço de Trabalho' */
{
  int_T currIndex = rtDW.FromWorkspace_IWORK.PrevIndex+1;
  creal_T *pDataValues = (creal_T *) rtDW.FromWorkspace_PWORK.DataPtr;
  if (currIndex >= 128) {
    currIndex = 0;
  }

  if (currIndex < 128) {
    pDataValues += currIndex;
    rtb_DopplerFrequencyShift = *pDataValues;
  } else {
    pDataValues += (127);
    rtb_DopplerFrequencyShift = *pDataValues;
  }

  rtDW.FromWorkspace_IWORK.PrevIndex = currIndex;
}

/* Produto: '<S5>/Produto' */
rtb_SineWave.re = rtb_ComplectoRealImag_o2
 * rtb_DopplerFrequencyShift.re;
rtb_SineWave.im = rtb_ComplectoRealImag_o2
 * rtb_DopplerFrequencyShift.im;

/* Ganho: '<S83>/ScaleInput' */
rtDW.ScaleInput.re = 14.142135623730951
 * rtb_SineWave.re;
rtDW.ScaleInput.im = 14.142135623730951
 * rtb_SineWave.im;

```

```

/* DiscreteFir: '<S90>/RFfilt' */
/* ConSomair linha de atraso e início de amostras de entrada
*/
rtb_SomaofElements_re = 0.0;
rtb_SomaofElements_im = 0.0;
j = 0;
while (j
      < 1) {
    rtb_SomaofElements_re
      += rtDW.ScaleInput.re
        * rtConstP.pooled15[0L];
    rtb_SomaofElements_im
      += rtDW.ScaleInput.im
        * rtConstP.pooled15[0L];
    j = 1;
}

para (j = 0; j
      < 127; j
      ++) {
    rtb_SomaofElements_re
      += rtConstP.pooled15[(int32_T)(1
        + j)]
        * rtDW.RFfilt_states[(int32_T)j].re;
    rtb_SomaofElements_im
      += rtConstP.pooled15[(int32_T)(1
        + j)]
        * rtDW.RFfilt_states[(int32_T)j].im;
}

RFfilt.re = rtb_SomaofElements_re;
RFfilt.im = rtb_SomaofElements_im;

/* Atualizar linha de atraso para o próximo quadro
*/
para (j = 125; j
      >= 0; j
      --) {
    rtDW.RFfilt_states[(int32_T)(1
      + j)] = rtDW.RFfilt_states[(int32_T)j];
}

rtDW.RFfilt_states[0L] = rtDW.ScaleInput;

/* SwitchCase: '<S91>/Switch Case' incorporados:
* Ganho: '<S91>/Ganho'
* Entrada: '<S101>/Entrada'
*/
switch ((int32_T)(0.0
                * rtDW.Sonda_o)) {
case 0L:
    /* Saída para o subsistema IfAction: '<S91>/y_eq_u' incorporados:
    * ActionPort: '<S101>/Porta de ação'
    */
    rtDW.Merge = RFfilt;

```

```

/* Fim das saídas para o subsistema: '<S91>/y_eq_u' */
break;

case 1L:
/* Saídas do subsistema Outputs: '<S91>/amonly' incorporados:
* ActionPort: '<S97>/Action Port'
*/
/* ComplexToMagnitudeAngle: '<S97>/Complexo para Magnitude-Angle1'
incorporados:
* DiscreteFir: '<S90>/Rffilt'
*/
rtb_ComplextoRealImag_o2 = rt_hypotd_snf(rtb_SomaofElements_re,
rtb_SomaofElements_im);
rtb_ComplextoMagnitudeAngle1__j = rt_atan2d_snf(rtb_SomaofElements_im,
rtb_SomaofElements_re);

/* Lookup: '<S106>/Look-Up Table'
* Sobre '<S106>/Look-Up Table':
* Input0 Tipo de Dado: Ponto Flutuante real_T
* Output0 Tipo de Dado: Ponto Flutuante real_T
* Método Lookup: Linear_Endpoint
*
* XData está alinhado e uniparamemente espaçado, então o algoritmo só
precisa
* o valor do primeiro elemento, o último elemento e o espaçamento.
* Por eficiência, XData é excluído do código gerado.
* O parâmetro YData usa o mesmo tipo de dados e escala como Output0
*/
LookupEven_real_T_real_T( &(rtb_LookUpTable_f), rtConstP.pooled6,
rtb_ComplextoRealImag_o2, 1.0, 1U, 1.0);

/* Ganho: '<S97>/Output Ganho' incorporados:
* MagnitudeAngleToComplex: '<S97>/Magnitude-Angle to Complex1'
*/
rtDW.Merge.re = rtb_LookUpTable_f
* cos(rtb_ComplextoMagnitudeAngle1__j);
rtDW.Merge.im = rtb_LookUpTable_f
* sin(rtb_ComplextoMagnitudeAngle1__j);

/* Saídas do subsistema Outputs: '<S91>/amonly' */
break;

case 2L:
/* Saídas do subsistema IfAction: '<S91>/ampm' incorporados:
* ActionPort: '<S98>/Action Port'
*/
/* ComplexToMagnitudeAngle: '<S98>/Complexo para Magnitude-Angle1'
incorporados:
* DiscreteFir: '<S90>/Rffilt'
*/
rtb_ComplextoRealImag_o2 = rt_hypotd_snf(rtb_SomaofElements_re,
rtb_SomaofElements_im);

/* Lookup: '<S107>/Look-Up Table'
* Sobre '<S107>/Look-Up Table':
* Input0 Tipo de Dado: Ponto Flutuante real_T
* Output0 Tipo de Dado: Ponto Flutuante real_T

```

```

    * Método Lookup: Linear_Endpoint
    *
    * XData está alinhado e uniparamemente espaçado, então o algoritmo só
precisa
    * o valor do primeiro elemento, o último elemento e o espaçamento
    * Para eficiência, o XData é excluído do código gerado.
    * O parâmetro YData usa o mesmo tipo de dados e escala como Output0
    */
    LookupEven_real_T_real_T( &(rtb_LookUpTable_a), rtConstP.pooled6,
        rtb_ComplextoRealImag_o2, 1.0, 1U, 1.0);

    /* Soma: '<S98>/Soma1' incorporados:
    * ComplexToMagnitudeAngle: '<S98>/Complexo para Magnitude-Angle1'
    * DiscreteFir: '<S90>/Rffilt'
    * Lookup: '<S108>/Look-Up Table'
    */
    rtb_ComplextoRealImag_o2 = rt_Lookup(rtConstP.pooled6, 2,
        rtb_ComplextoRealImag_o2, rtConstP.pooled6)
        + rt_atan2d_snf(rtb_SomaofElements_im, rtb_SomaofElements_re);

    /* Ganho: '<S98>/Output Ganho' incorporados:
    * MagnitudeAngleToComplex: '<S98>/Angulo e Magnitude Complex1'
    */
    rtDW.Merge.re = rtb_LookUpTable_a
        * cos(rtb_ComplextoRealImag_o2);
    rtDW.Merge.im = rtb_LookUpTable_a
        * sin(rtb_ComplextoRealImag_o2);

    /* Fim das Saídas para o Subsistema: '<S91>/ampm' */
break;

case 3L:
    /* Saída para Subsistema IfAction: '<S91>/poly' incorporados:
    * ActionPort: '<S100>/Action Port'
    */
    /* ComplexToMagnitudeAngle: '<S100>/Complexo para Magnitude-Angle1'
    incorporados:
    * DiscreteFir: '<S90>/Rffilt'
    */
    rtb_ComplextoRealImag_o2 = rt_hypotd_snf(rtb_SomaofElements_re,
        rtb_SomaofElements_im);
    rtb_ComplextoMagnitudeAngle1__j = rt_atan2d_snf(rtb_SomaofElements_im,
        rtb_SomaofElements_re);

    /* Saturação: '<S114>/Saturação de Entrada' */
if (rtb_ComplextoRealImag_o2
        <= 0.0) {
        rtb_ComplextoRealImag_o2 = 0.0;
    }

    /* Fim da Saturação: '<S114>/Saturação de Entrada' */

    /* Soma: '<S114>/Soma' incorporados:
    * Constantee: '<S114>/Constante'
    * Constantee: '<S114>/Constante1'
    * Constantee: '<S114>/Constante2'
    * Ganho: '<S114>/c3'

```



```

* Ganho: '<S114>/c5'
* Ganho: '<S114>/c7'
* Matemática: '<S114>/Matemática Function'
* Matemática: '<S114>/Matemática Function1'
* Matemática: '<S114>/Matemática Function2'
*/
rtb_ComplextoRealImag_o2 = ((0.0
* rt_powd_snf(rtb_ComplextoRealImag_o2, 3.0)
+ rtb_ComplextoRealImag_o2)
+ 0.0
* rt_powd_snf(rtb_ComplextoRealImag_o2, 5.0))
+ 0.0
* rt_powd_snf(rtb_ComplextoRealImag_o2, 7.0);

/* Ganho: '<S100>/Output Ganho' incorporados:
* MagnitudeAngleToComplex: '<S100>/Magnitude-Angle para Complex1'
*/
rtDW.Merge.re = rtb_ComplextoRealImag_o2
* cos(rtb_ComplextoMagnitudeAngle1__j);
rtDW.Merge.im = rtb_ComplextoRealImag_o2
* sin(rtb_ComplextoMagnitudeAngle1__j);

/* Fim das saídas para SubSystem: '<S91>/poly' */
break;

case 4L:
/* Saídas para IfAction SubSystem: '<S91>/p2d' incorporados:
* ActionPort: '<S99>/Action Port'
*/
p2d(RFfilt, &rtDW.Merge, (ConstB_p2d *)&rtConstB.p2d_d, &rtDW.p2d_d);

/* Fim das saídas para SubSystem: '<S91>/p2d' */
break;
}

/* Fim do SwitchCase: '<S91>/Switch Case' */

/* SwitchCase: '<S96>/Switch Case' incorporados:
* Ganho: '<S96>/Ganho'
* Importar: '<S102>/In'
*/
switch ((int32_T)(0.0
* rtDW.Sonda_f)) {
case 0L:
/* Saídas para IfAction SubSystem: '<S96>/nophasenoise' incorporados:
* ActionPort: '<S102>/Action Port'
*/
rtDW.Merge_j = rtDW.Merge;

/* Fim das Saídas para SubSystem: '<S96>/nophasenoise' */
break;

case 1L:
/* Saídas para IfAction SubSystem: '<S96>/phasenoise' incorporados:
* ActionPort: '<S103>/Action Port'
*/
phasenoise(rtDW.Merge, &rtDW.Merge_j, &rtDW.phasenoise_k);

```

```

    /* Fim das Saídas para SubSystem: '<S96>/phasenoise' */
    break;
}

/* Fim para SwitchCase: '<S96>/Switch Case' */

/* RealImagtoComplex: '<S95>/Real-Imag para Complex' incorporados:
 * ComplexToRealImag: '<S95>/Complex para Real-Imag'
 */
rtDW.RealImagtoComplex = rtDW.Merge_j;

/* DiscreteFir: '<S92>/RFfilt' */
/* ConSomair linha de atraso e início de amostras de entrada
*/
rtb_SomaofElements_re = 0.0;
rtb_SomaofElements_im = 0.0;
j = 0;
while (j
    < 1) {
    rtb_SomaofElements_re
    += rtDW.RealImagtoComplex.re
    * rtConstP.RFfilt_Coefficients_g[0L].re
    - rtDW.RealImagtoComplex.im
    * rtConstP.RFfilt_Coefficients_g[0L].im;
    rtb_SomaofElements_im
    += rtDW.RealImagtoComplex.re
    * rtConstP.RFfilt_Coefficients_g[0L].im
    + rtDW.RealImagtoComplex.im
    * rtConstP.RFfilt_Coefficients_g[0L].re;
    j = 1;
}

para (j = 0; j
    < 127; j
    ++) {
    rtb_SomaofElements_re
    += rtConstP.RFfilt_Coefficients_g[(int32_T)(1
    + j)].re
    * rtDW.RFfilt_states_l[(int32_T)j].re
    - rtConstP.RFfilt_Coefficients_g[(int32_T)(1
    + j)].im
    * rtDW.RFfilt_states_l[(int32_T)j].im;
    rtb_SomaofElements_im
    += rtConstP.RFfilt_Coefficients_g[(int32_T)(1
    + j)].im
    * rtDW.RFfilt_states_l[(int32_T)j].re
    + rtConstP.RFfilt_Coefficients_g[(int32_T)(1
    + j)].re
    * rtDW.RFfilt_states_l[(int32_T)j].im;
}

RFfilt_f.re = rtb_SomaofElements_re;
RFfilt_f.im = rtb_SomaofElements_im;

/* Atualizar linha de atraso para o próximo quadro */
para (j = 125; j

```

```

    >= 0; j
    --) {
rtDW.RFfilt_states_l[(int32_T)(1
    + j)] = rtDW.RFfilt_states_l[(int32_T)j];
}

rtDW.RFfilt_states_l[0L] = rtDW.RealImagtoComplex;

/* SwitchCase: '<S93>/Switch Case' incorporados:
 * Ganho: '<S93>/Ganho'
 * Impotação: '<S121>/In'
 */
switch ((int32_T)(3.0
    * rtDW.Sonda_k)) {
case 0L:
    /* Saídas para IfAction SubSystem: '<S93>/y_eq_u' incorporados:
     * ActionPort: '<S121>/Action Port'
     */
    rtDW.Merge_c = RFfilt_f;

    /* Fim das Saídas para SubSystem: '<S93>/y_eq_u' */
    break;

case 1L:
    /* Saídas para IfAction SubSystem: '<S93>/amonly' incorporados:
     * ActionPort: '<S117>/Action Port'
     */
    /* ComplexToMagnitudeAngle: '<S117>/Complex para Magnitude-Angle1'
incorporados:
     * DiscreteFir: '<S92>/RFfilt'
     */
    rtb_ComplextoRealImag_o2 = rt_hypotd_snf(rtb_SomaofElements_re,
        rtb_SomaofElements_im);
    rtb_ComplextoMagnitudeAngle1__j = rt_atan2d_snf(rtb_SomaofElements_im,
        rtb_SomaofElements_re);

    /* Lookup: '<S126>/Look-Up Table'
     * Sobre '<S126>/Look-Up Table':
     * Input0 Tipo de Dado: Ponto Flutuante real_T
     * Output0 Tipo de Dado: Ponto Flutuante real_T
     * Método Lookup: Linear_Endpoint
     */
    * XData está alinhado e uniparamemente espaçado, então o algoritmo só precisa
     * o valor do primeiro elemento, o último elemento e o espaçamento.
     * Para efficiency, XData é excluído do código gerado.
     * YData parâmetro usa o mesmo tipo de dados e escala como Output0
     */
    LookupEven_real_T_real_T( &(rtb_LookUpTable_j), rtConstP.pooled6,
        rtb_ComplextoRealImag_o2, 1.0, 1U, 1.0);

    /* Ganho: '<S117>/Output Ganho' incorporados:
     * MagnitudeAngleToComplex: '<S117>/Magnitude-Angle para Complex1'
     */
    rtDW.Merge_c.re = rtb_LookUpTable_j
        * cos(rtb_ComplextoMagnitudeAngle1__j)
        * 0.01;
    rtDW.Merge_c.im = rtb_LookUpTable_j

```

```

    * sin(rtb_ComplextoMagnitudeAngle1__j)
    * 0.01;

/* Fim das Saídas para SubSystem: '<S93>/amonly' */
break;

case 2L:
/* Saídas para IfAction SubSystem: '<S93>/ampm' incorporados:
 * ActionPort: '<S118>/Action Port'
 */
/* ComplexToMagnitudeAngle: '<S118>/Complex para Magnitude-Angle1'
incorporados:
 * DiscreteFir: '<S92>/RFfilt'
 */
rtb_ComplextoRealImag_o2 = rt_hypotd_snf(rtb_SomaofElements_re,
    rtb_SomaofElements_im);

/* Lookup: '<S127>/Look-Up Table'
 * Sobre '<S127>/Look-Up Table':
 * Input0 Tipo de Dado: Ponto Flutuante real_T
 * Output0 Tipo de Dado: Ponto Flutuante real_T
 * Método Lookup: Linear_Endpoint
 *
 * XData está alinhado e uniparamemente espaçado, então o algoritmo só
precisa
 * o valor do primeiro elemento, o último elemento e o espaçamento.
 * Para eficiência, o XData é excluído do código gerado.
 * O parâmetro YData usa o mesmo tipo de dados e escala como Output0
 * /
LookupEven_real_T_real_T( &(rtb_LookUpTable_k), rtConstP.pooled6,
    rtb_ComplextoRealImag_o2, 1.0, 1U, 1.0);

/* Soma: '<S118>/Soma1' incorporados:
 * ComplexToMagnitudeAngle: '<S118>/Complex para Magnitude-Angle1'
 * DiscreteFir: '<S92>/RFfilt'
 * Lookup: '<S128>/Look-Up Table'
 */
rtb_ComplextoRealImag_o2 = rt_Lookup(rtConstP.pooled6, 2,
    rtb_ComplextoRealImag_o2, rtConstP.pooled6)
    + rt_atan2d_snf(rtb_SomaofElements_im, rtb_SomaofElements_re);

/* Ganho: '<S118>/Output Ganho' incorporados:
 * MagnitudeAngleToComplex: '<S118>/Magnitude-Angle para Complex1'
 */
rtDW.Merge_c.re = rtb_LookUpTable_k
    * cos(rtb_ComplextoRealImag_o2)
    * 0.01;
rtDW.Merge_c.im = rtb_LookUpTable_k
    * sin(rtb_ComplextoRealImag_o2)
    * 0.01;

/* Fim das Saídas para SubSystem: '<S93>/ampm' */
break;

case 3L:
/* Saídas para IfAction SubSystem: '<S93>/poly' incorporados:
 * ActionPort: '<S120>/Action Port'

```

```

*/
/* ComplexToMagnitudeAngle: '<S120>/Complex para Magnitude-Angle1'
incorporados:
* DiscreteFir: '<S92>/RFfilt'
*/
rtb_ComplextoRealImag_o2 = rt_atan2d_snf(rtb_SomaofElements_im,
    rtb_SomaofElements_re);
rtb_ComplextoMagnitudeAngle1__j = rt_hypotd_snf(rtb_SomaofElements_re,
    rtb_SomaofElements_im);

/* Saturação: '<S134>/Entrada saturada' */
if (rtb_ComplextoMagnitudeAngle1__j > 0.408248290463863) {
    rtb_ComplextoMagnitudeAngle1__j = 0.408248290463863;
}

/* Fim da Saturação: '<S134>/Entrada saturada' */

/* Matemática: '<S134>/Matemática Function' incorporados:
* Constantee: '<S134>/Constante'
*/
rtb_SomaofElements_re = rt_powd_snf(rtb_ComplextoMagnitudeAngle1__j,
3.0);

/* Matemática: '<S134>/Matemática Function1' incorporados:
* Constantee: '<S134>/Constante1'
*/
rtb_SomaofElements_im = rt_powd_snf(rtb_ComplextoMagnitudeAngle1__j,
5.0);

/* Matemática: '<S134>/Matemática Function2' incorporados:
* Constantee: '<S134>/Constante2'
*/
rtb_ComplextoMagnitudeAngle1_n1 = rt_powd_snf
    (rtb_ComplextoMagnitudeAngle1__j, 7.0);

/* Ganho: '<S120>/Output Ganho' incorporados:
* Ganho: '<S134>/c1'
* Ganho: '<S134>/c3'
* Ganho: '<S134>/c5'
* Ganho: '<S134>/c7'
* MagnitudeAngleToComplex: '<S120>/Magnitude-Angle para Complex1'
* Soma: '<S134>/Soma'
*/
rtDW.Merge_c.re = (((100.0
    * rtb_ComplextoMagnitudeAngle1__j
    + -200.0
    * rtb_SomaofElements_re)
    + 0.0
    * rtb_SomaofElements_im)
    + 0.0
    * rtb_ComplextoMagnitudeAngle1_n1)
    * cos(rtb_ComplextoRealImag_o2)
    * 0.01;
rtDW.Merge_c.im = (((100.0
    * rtb_ComplextoMagnitudeAngle1__j
    + -200.0
    * rtb_SomaofElements_re)

```

```

        + 0.0
        * rtb_SomaofElements_im)
    + 0.0
    * rtb_ComplextoMagnitudeAngle1_n1)
    * sin(rtb_ComplextoRealImag_o2)
    * 0.01;

    /* Fim das Saídas para SubSystem: '<S93>/poly' */
    break;

case 4L:
    /* Saídas para IfAction SubSystem: '<S93>/p2d' incorporados:
    * ActionPort: '<S119>/Action Port'
    */
    p2d(RFfilt_f, &rtDW.Merge_c, (ConstB_p2d *)&rtConstB.p2d_b,
    &rtDW.p2d_b);

    /* Fim das Saídas para SubSystem: '<S93>/p2d' */
    break;
}

/* Fim do SwitchCase: '<S93>/Switch Case' */

/* SwitchCase: '<S116>/Switch Case' incorporados:
* Ganho: '<S116>/Ganho'
* Importação: '<S122>/In'
*/
switch ((int32_T)(0.0
            * rtDW.Sonda)) {
case 0L:
    /* Saídas para IfAction SubSystem: '<S116>/nophasenoise' incorporados:
    * ActionPort: '<S122>/Action Port'
    */
    rtDW.Merge_cm = rtDW.Merge_c;

    /* Fim das Saídas para SubSystem: '<S116>/nophasenoise' */
    break;

case 1L:
    /* Saídas para IfAction SubSystem: '<S116>/phasenoise' incorporados:
    * ActionPort: '<S123>/Action Port'
    */
    phasenoise(rtDW.Merge_c, &rtDW.Merge_cm, &rtDW.phasenoise_o);

    /* Fim das Saídas paraSubSystem: '<S116>/phasenoise' */
    break;
}

/* Fim do SwitchCase: '<S116>/Switch Case' */

/* RealImagToComplex: '<S115>/Real-Imag para Complex' incorporados:
* ComplexToRealImag: '<S115>/Complex para Real-Imag'
*/
rtDW.RealImagtoComplex = rtDW.Merge_cm;

/* DiscreteFir: '<S94>/RFfilt' */
/* ConSomair linha de atraso e início de amostras de entrada */

```

```

rtb_SomaofElements_re = 0.0;
rtb_SomaofElements_im = 0.0;
j = 0;
while (j
    < 1) {
    rtb_SomaofElements_re
    += rtDW.RealImagtoComplex.re
    * rtConstP.RFfilt_Coefficients[0L];
    rtb_SomaofElements_im
    += rtDW.RealImagtoComplex.im
    * rtConstP.RFfilt_Coefficients[0L];
    j = 1;
}

para (j = 0; j
    < 127; j
    ++) {
    rtb_SomaofElements_re
    += rtConstP.RFfilt_Coefficients[(int32_T)(1
    + j)]
    * rtDW.RFfilt_states_p[(int32_T)j].re;
    rtb_SomaofElements_im
    += rtConstP.RFfilt_Coefficients[(int32_T)(1
    + j)]
    * rtDW.RFfilt_states_p[(int32_T)j].im;
}

rtb_SineWave.re = rtb_SomaofElements_re;
rtb_SineWave.im = rtb_SomaofElements_im;

/* Atualizar linha de atraso para o próximo quadro */
para (j = 125; j
    >= 0; j
    --) {
    rtDW.RFfilt_states_p[(int32_T)(1
    + j)] = rtDW.RFfilt_states_p[(int32_T)j];
}

rtDW.RFfilt_states_p[0L] = rtDW.RealImagtoComplex;

/* Fim do DiscreteFir: '<S94>/RFfilt' */

/* S-Function (sdsprandsrc2): '<S135>/Random Source' */
RandSrc_GZ_Z(&rtb_DopplerFrequencyShift, &rtConstP.pooled16, 1L,
    &rtConstP.pooled2, 1L, rtDW.RandomSource_STATE_DWORK, 1L,
1L);

/* Produto: '<S135>/Produto' */
RFfilt = rtb_DopplerFrequencyShift;

/* DSP System Toolbox Conversão do Status do Quadro (sdspfrmconv) -
<S136>/Inherit */
rtDW.RealImagtoComplex = RFfilt;

/* DiscreteFir: '<S87>/RFfilt' */
RFfilt.re = rtDW.RealImagtoComplex.re
    * 0.0;

```

```

Rffilt.im = rtDW.RealImagtoComplex.im
* 0.0;

/* Soma: '<S83>/Soma' */
rtb_SineWave.re
+= Rffilt.re;
rtb_SineWave.im
+= Rffilt.im;

/* Ganho: '<S83>/ScaleOutput' */
rtb_SineWave.re
*= 0.1414213562373095;
rtb_SineWave.im
*= 0.1414213562373095;

/* Ganho: '<S12>/Ganho' */
rtb_SineWave.re
*= 1.4125375446227544;
rtb_SineWave.im
*= 1.4125375446227544;

/* Ganho: '<S4>/Ganho' incorporados:
* Soma: '<S3>/Soma dos Elementos'
*/
rtDW.Ganho.re = 5.5096364425368848E-12
* rtb_SineWave.re;
rtDW.Ganho.im = 5.5096364425368848E-12
* rtb_SineWave.im;

/* Fcn: '<S1>/ Calcular o atraso das amostras à distância' */
rtb_ComplextoRealImag_o2 = rtb_RadarSignalDegradation
/ 300000000.0
/ 1.5625E-5;

/* S-Function (sdspvdly2): '<S1>/Retorno Atraso' */
rtb_ComplextoMagnitudeAngle1__j = fabs(rtb_ComplextoRealImag_o2);
if (rtb_ComplextoMagnitudeAngle1__j < 4.503599627370496E+15) {
    if (rtb_ComplextoMagnitudeAngle1__j >= 0.5) {
        rtb_ComplextoRealImag_o2 = floor(rtb_ComplextoRealImag_o2 + 0.5);
    } else {
        rtb_ComplextoRealImag_o2
        *= 0.0;
    }
}

if (rtb_ComplextoRealImag_o2
< 2.147483648E+9) {
    if (rtb_ComplextoRealImag_o2
>= -2.147483648E+9) {
        k = (int32_T)rtb_ComplextoRealImag_o2;
    } else {
        k = MIN_int32_T;
    }
} else {
    k = MAX_int32_T;
}

```



```

if (k
    < 0L) {
    k = 0L;
} else {
    if ((uint32_T)k
        > 100UL) {
        k = 100L;
    }
}

if ((int16_T)k
    == 0) {
    RFfilt = rtDW.Ganho;
} else {
    k = rtDW.DelayReturn_BUFF_OFFSET
        - k;
    if (k
        < 0L) {
        k
            += 101L;
    }

    RFfilt = rtDW.DelayReturn_BUFF[k];
}

/* Fim do S-Function (sdspvdly2): '<S1>/Retorno Atrasado' */

/* S-Function (sdsp sine2): '<S1>/Onda Senoidal' */
rtb_SineWave.re = cos(rtDW.SineWave_AccFreqNorm);
rtb_SineWave.im = sin(rtDW.SineWave_AccFreqNorm);

/* Atualizar o valor de frequência normalizada acumulada
para a próxima amostra. Mantenha no alcance [0 2*pi) */
rtDW.SineWave_AccFreqNorm
    += 0.0042509675593886886;
if (rtDW.SineWave_AccFreqNorm
    >= 6.2831853071795862) {
    rtDW.SineWave_AccFreqNorm
        -= 6.2831853071795862;
} else {
    if (rtDW.SineWave_AccFreqNorm
        < 0.0) {
        rtDW.SineWave_AccFreqNorm
            += 6.2831853071795862;
    }
}

/* Fim do S-Function (sdsp sine2): '<S1>/Onda Senoidal' */

/* Produto: '<S1>/Doppler Mudança de Frequência' */
rtb_DopplerFrequencyShift.re = RFfilt.re
    * rtb_SineWave.re
    - RFfilt.im
    * rtb_SineWave.im;
rtb_DopplerFrequencyShift.im = RFfilt.re
    * rtb_SineWave.im
    + RFfilt.im

```

```

* rtb_SineWave.re;

/* Fcn: '<S10>/Degradação do Sinal' */
rtb_ComplextoRealImag_o2 = 1.0
/ rt_powd_snf(rtb_RadarSignalDegradation, 4.0);
if (rtb_ComplextoRealImag_o2
    < 0.0) {
    rtb_RadarSignalDegradation =
        -sqrt(
            -rtb_ComplextoRealImag_o2);
} else {
    rtb_RadarSignalDegradation = sqrt(rtb_ComplextoRealImag_o2);
}

/* Fim do Fcn: '<S10>/ Degradação do Sinal ' */

/* Produto: '<S10>/Produto' */
Rffilt.re = rtb_DopplerFrequencyShift.re
* rtb_RadarSignalDegradation;
Rffilt.im = rtb_DopplerFrequencyShift.im
* rtb_RadarSignalDegradation;

/* Ganho: '<S10>/Potência de Transmissão' */
Rffilt.re
*= 0.015553157227930131;
Rffilt.im
*= 0.015553157227930131;

/* Ganho: '<S11>/Ganho' */
Rffilt.re
*= 3.1622776601683795;
Rffilt.im
*= 3.1622776601683795;

/* Soma: '<S2>/Soma dos Elementos' */
rtb_SomaofElements_re = Rffilt.re;
rtb_SomaofElements_im = Rffilt.im;

/* S-Function (sdsprandsrc2): '<S74>/Fonte Aleatória' */
RandSrc_GZ_Z(&Rffilt, &rtConstP.pooled16, 1L, &rtConstP.pooled2, 1L,
    rtDW.RandomSource_STATE_DWORK_m, 1L, 1L);

/* Ganho: '<S19>/Ganho' incorporados:
* Produto: '<S19>/Produto'
*/
rtb_ComplextoRealImag_o2 = rtDW.Sonda_o1
/ rtDW.Sonda_o2[0]
* 4.003885E-21;

/* S-Function (scomawgnchan2): '<S74>/AWGN Dinâmico' */
rtb_ComplextoMagnitudeAngle1__j = sqrt(rtb_ComplextoRealImag_o2);
rtDW.RealImagtoComplex.re = rtb_ComplextoMagnitudeAngle1__j
* Rffilt.re;
rtDW.RealImagtoComplex.im = rtb_ComplextoMagnitudeAngle1__j
* Rffilt.im;
rtDW.RealImagtoComplex.re
+= rtb_SomaofElements_re;

```

```

rtDW.RealImagtoComplex.im
    += rtb_SomaofElements_im;

/* DiscreteFir: '<S27>/RFfilt' */
/* Linha de atraso ConSomae e inicio de amostras de entrada */
rtb_SomaofElements_re = 0.0;
rtb_SomaofElements_im = 0.0;
j = 0;
while (j
    < 1) {
    rtb_SomaofElements_re
    += rtDW.RealImagtoComplex.re
    * rtConstP.RFfilt_Coefficients_m[0L].re
    - rtDW.RealImagtoComplex.im
    * rtConstP.RFfilt_Coefficients_m[0L].im;
    rtb_SomaofElements_im
    += rtDW.RealImagtoComplex.re
    * rtConstP.RFfilt_Coefficients_m[0L].im
    + rtDW.RealImagtoComplex.im
    * rtConstP.RFfilt_Coefficients_m[0L].re;
    j = 1;
}

para (j = 0; j
    < 127; j
    ++) {
    rtb_SomaofElements_re
    += rtConstP.RFfilt_Coefficients_m[(int32_T)(1
    + j)].re
    * rtDW.RFfilt_states_a[(int32_T)j].re
    - rtConstP.RFfilt_Coefficients_m[(int32_T)(1
    + j)].im
    * rtDW.RFfilt_states_a[(int32_T)j].im;
    rtb_SomaofElements_im
    += rtConstP.RFfilt_Coefficients_m[(int32_T)(1
    + j)].im
    * rtDW.RFfilt_states_a[(int32_T)j].re
    + rtConstP.RFfilt_Coefficients_m[(int32_T)(1
    + j)].re
    * rtDW.RFfilt_states_a[(int32_T)j].im;
}

RFfilt_i.re = rtb_SomaofElements_re;
RFfilt_i.im = rtb_SomaofElements_im;

/* Atualizar linha de atraso para o próximo quadro */
para (j = 125; j
    >= 0; j
    --) {
    rtDW.RFfilt_states_a[(int32_T)(1
    + j)] = rtDW.RFfilt_states_a[(int32_T)j];
}

rtDW.RFfilt_states_a[0L] = rtDW.RealImagtoComplex;

/* SwitchCase: '<S28>/Switch Case' incorporados:
* Ganho: '<S28>/Ganho'

```

```

* Importação: '<S38>/In'
*/
switch ((int32_T)(3.0
          * rtDW.Sonda_fd)) {
case 0L:
/* Saídas paraIfAction SubSystem: '<S28>/y_eq_u' incorporados:
 * ActionPort: '<S38>/Action Port'
 */
rtDW.Merge_i = RFfilt_i;

/* Fim das Saídas paraSubSystem: '<S28>/y_eq_u' */
break;

case 1L:
/* Saídas paraIfAction SubSystem: '<S28>/amonly' incorporados:
 * ActionPort: '<S34>/Action Port'
 */
/* ComplexToMagnitudeAngle: '<S34>/Complex para Magnitude-Angle1'
incorporados:
 * DiscreteFir: '<S27>/RFfilt'
 */
rtb_ComplextorealImag_o2 = rt_hypotd_snf(rtb_SomaofElements_re,
    rtb_SomaofElements_im);
rtb_ComplextomagnitudeAngle1__j = rt_atan2d_snf(rtb_SomaofElements_im,
    rtb_SomaofElements_re);

/* Lookup: '<S43>/Look-Up Table'
 * Sobre '<S43>/Look-Up Table':
 * Input0 Tipo de Dado: Ponto Flutuante real_T
 * Output0 Tipo de Dado: Ponto Flutuante real_T
 * Método Lookup: Linear_Endpoint
 *
 * XData está alinhado e uniparamemente espaçado, então o algoritmo só
precisa
 * o valor do primeiro elemento, o último elemento e o espaçamento.
 * Para eficiência, o XData é excluído do código gerado.
 * O parâmetro YData usa o mesmo tipo de dados e escala como Output0 */
LookupEven_real_T_real_T( &(rtb_LookUpTable_kw), rtConstP.pooled6,
    rtb_ComplextorealImag_o2, 1.0, 1U, 1.0);

/* Ganho: '<S34>/Output Ganho' incorporados:
 * MagnitudeAngleToComplex: '<S34>/Magnitude-Angle para Complex1'
 */
rtDW.Merge_i.re = rtb_LookUpTable_kw
    * cos(rtb_ComplextomagnitudeAngle1__j)
    * 2.0;
rtDW.Merge_i.im = rtb_LookUpTable_kw
    * sin(rtb_ComplextomagnitudeAngle1__j)
    * 2.0;

/* Fim das Saídas paraSubSystem: '<S28>/amonly' */
break;

case 2L:
/* Saídas paraIfAction SubSystem: '<S28>/ampm' incorporados:
 * ActionPort: '<S35>/Action Port'
 */

```

```

/* ComplexToMagnitudeAngle: '<S35>/Complex para Magnitude-Angle1'
incorporados:
* DiscreteFir: '<S27>/RFfilt'
*/
rtb_ComplextoRealImag_o2 = rt_hypotd_snf(rtb_SomaofElements_re,
rtb_SomaofElements_im);

/* Lookup: '<S44>/Look-Up Table'
* Sobre '<S44>/Look-Up Table':
* Input0 Tipo de Dado: Ponto Flutuante real_T
* Output0 Tipo de Dado: Ponto Flutuante real_T
* Método Lookup: Linear_Endpoint
*
* XData está alinhado e uniparamemente espaçado, então o algoritmo só
precisa
* o valor do primeiro elemento, o último elemento e o espaçamento.
* Para eficiência, o XData é excluído do código gerado.
* O parâmetro YData usa o mesmo Tipo de Dado e é dimensionado como Output0
*/
LookupEven_real_T_real_T( &(rtb_LookUpTable_o), rtConstP.pooled6,
rtb_ComplextoRealImag_o2, 1.0, 1U, 1.0);

/* Soma: '<S35>/Soma1' incorporados:
* ComplexToMagnitudeAngle: '<S35>/Complex para Magnitude-Angle1'
* DiscreteFir: '<S27>/RFfilt'
* Lookup: '<S45>/Look-Up Tabela'
*/
rtb_ComplextoRealImag_o2 = rt_Lookup(rtConstP.pooled6, 2,
rtb_ComplextoRealImag_o2, rtConstP.pooled6)
+ rt_atan2d_snf(rtb_SomaofElements_im, rtb_SomaofElements_re);

/* Ganho: '<S35>/Output Ganho' incorporados:
* MagnitudeAngleToComplex: '<S35>/Magnitude-Angle para Complex1'
*/
rtDW.Merge_i.re = rtb_LookUpTable_o
* cos(rtb_ComplextoRealImag_o2)
* 2.0;
rtDW.Merge_i.im = rtb_LookUpTable_o
* sin(rtb_ComplextoRealImag_o2)
* 2.0;

/* Fim das Saídas paraSubSystem: '<S28>/ampm' */
break;

case 3L:
/* Saídas paraIfAction SubSystem: '<S28>/poly' incorporados:
* ActionPort: '<S37>/Action Port'
*/
/* ComplexToMagnitudeAngle: '<S37>/Complex para Magnitude-Angle1'
incorporados:
* DiscreteFir: '<S27>/RFfilt'
*/
rtb_ComplextoRealImag_o2 = rt_atan2d_snf(rtb_SomaofElements_im,
rtb_SomaofElements_re);
rtb_ComplextoMagnitudeAngle1__j = rt_hypotd_snf(rtb_SomaofElements_re,
rtb_SomaofElements_im);

```

```

/* Saturar: '<S51>/Input saturation' */
if (rtb_ComplextoMagnitudeAngle1__j > 25819.888974716112) {
    rtb_ComplextoMagnitudeAngle1__j = 25819.888974716112;
}

/* Fim do Saturar: '<S51>/Input saturation' */

/* Matemática: '<S51>/Matemática Function' incorporados:
 * Constante: '<S51>/Constante'
 */
rtb_SomaofElements_re = rt_powd_snf(rtb_ComplextoMagnitudeAngle1__j,
3.0);

/* Matemática: '<S51>/Matemática Function1' incorporados:
 * Constante: '<S51>/Constante1'
 */
rtb_SomaofElements_im = rt_powd_snf(rtb_ComplextoMagnitudeAngle1__j,
5.0);

/* Matemática: '<S51>/Matemática Function2' incorporados:
 * Constante: '<S51>/Constante2'
 */
rtb_ComplextoMagnitudeAngle1_n1 = rt_powd_snf
(rtb_ComplextoMagnitudeAngle1__j, 7.0);

/* Ganho: '<S37>/Output Ganho' incorporados:
 * Ganho: '<S51>/c1'
 * Ganho: '<S51>/c3'
 * Ganho: '<S51>/c5'
 * Ganho: '<S51>/c7'
 * MagnitudeAngleToComplex: '<S37>/Magnitude-Angle para Complex1'
 * Soma: '<S51>/Soma'
 */
rtDW.Merge_i.re = (((0.5
    * rtb_ComplextoMagnitudeAngle1__j
    + -2.5E-10
    * rtb_SomaofElements_re)
+ 0.0
    * rtb_SomaofElements_im)
+ 0.0
    * rtb_ComplextoMagnitudeAngle1_n1)
    * cos(rtb_ComplextoRealImag_o2)
    * 2.0;
rtDW.Merge_i.im = (((0.5
    * rtb_ComplextoMagnitudeAngle1__j
    + -2.5E-10
    * rtb_SomaofElements_re)
+ 0.0
    * rtb_SomaofElements_im)
+ 0.0
    * rtb_ComplextoMagnitudeAngle1_n1)
    * sin(rtb_ComplextoRealImag_o2)
    * 2.0;

/* Fim das Saídas paraSubSystem: '<S28>/poly' */
break;

```

```

case 4L:
    /* Saídas paraIfAction SubSystem: '<S28>/p2d' incorporados:
     * ActionPort: '<S36>/Action Port'
     */
    p2d(RFfilt_i, &rtDW.Merge_i, (ConstB_p2d *)&rtConstB.p2d_e,
&rtDW.p2d_e);

    /* Fim das Saídas paraSubSystem: '<S28>/p2d' */
    break;
}

/* Fim do SwitchCase: '<S28>/Switch Case' */

/* SwitchCase: '<S33>/Switch Case' incorporados:
 * Inport: '<S39>/In'
 */
switch ((int32_T)rtDW.Sonda_g) {
case 0L:
    /* Saídas paraIfAction SubSystem: '<S33>/nophasenoise' incorporados:
     * ActionPort: '<S39>/Action Port'
     */
    rtDW.Merge_p = rtDW.Merge_i;

    /* Fim das Saídas paraSubSystem: '<S33>/nophasenoise' */
    break;

case 1L:
    /* Saídas paraIfAction SubSystem: '<S33>/phasenoise' incorporados:
     * ActionPort: '<S40>/Action Port'
     */
    /* S-Function (sdsprandsrc2): '<S41>/Random Source' */
    RandSrc_GZ_Z(&rtb_Inherit, &rtConstP.pooled16, 1L, &rtConstP.pooled2,
1L,
        rtDW.RandomSource_STATE_DWORK_h, 1L, 1L);

    /* Produto: '<S41>/Produto' */
    rtb_SineWave.re = rtb_Inherit.re
        * 1.4142135623730951;
    rtb_SineWave.im = rtb_Inherit.im
        * 1.4142135623730951;

    /* DSP System Toolbox Conversão do Status do Quadro (sdsprfmconv) -
    <S42>/Inherit */
    rtb_Inherit = rtb_SineWave;

    /* ComplexToRealImag: '<S40>/Complex para Real-Imag' */
    rtb_ComplextoRealImag_o2 = rtb_Inherit.re;

    /* DiscreteFir: '<S40>/Discrete FIR Filter' */
    /* ConSomae linha de atraso e início de amostras de entrada */
    rtb_ComplextoMagnitudeAngle1__j = 0.0;
    j = 0;
    while (j
        < 1) {
        rtb_ComplextoMagnitudeAngle1__j
            += rtb_ComplextoRealImag_o2
            * rtConstP.DiscreteFIRFilter_Coefficients[0L];
    }
}

```

```

    j = 1;
}

para (j = 0; j
    < 127; j
    ++) {
    rtb_ComplextoMagnitudeAngle1__j
    += rtConstP.DiscreteFIRFilter_Coefficients[(int32_T)(1
    + j)]
    * rtDW.DiscreteFIRFilter_states[(int32_T)j];
}

/* Atualizar linha de atraso para o próximo quadro */
para (j = 125; j
    >= 0; j
    --) {
    rtDW.DiscreteFIRFilter_states[(int32_T)(1
    + j)] = rtDW.DiscreteFIRFilter_states[(int32_T)j];
}

rtDW.DiscreteFIRFilter_states[0L] = rtb_ComplextoRealImag_o2;

/* Fim do DiscreteFir: '<S40>/Discrete FIR Filter' */

/* MagnitudeAngleToComplex: '<S40>/Magnitude-Angle para Complex' */
rtb_SineWave.re = cos(rtb_ComplextoMagnitudeAngle1__j);
rtb_SineWave.im = sin(rtb_ComplextoMagnitudeAngle1__j);

/* Produto: '<S40>/Produto' */
rtDW.Merge_p.re = rtDW.Merge_i.re
    * rtb_SineWave.re
    - rtDW.Merge_i.im
    * rtb_SineWave.im;
rtDW.Merge_p.im = rtDW.Merge_i.re
    * rtb_SineWave.im
    + rtDW.Merge_i.im
    * rtb_SineWave.re;

/* Fim das Saídas paraSubSystem: '<S33>/phasenoise' */
break;
}

/* Fim do SwitchCase: '<S33>/Switch Case' */

/* RealImagToComplex: '<S32>/Real-Imag para Complex' incorporados:
 * ComplexToRealImag: '<S32>/Complex para Real-Imag'
 */
rtDW.RealImagtoComplex = rtDW.Merge_p;

/* DiscreteFir: '<S29>/RFfilt' */
/* ConSomaefElements linha de atraso e inicio de amostras de entrada */
rtb_SomaofElements_re = 0.0;
rtb_SomaofElements_im = 0.0;
j = 0;
while (j
    < 1) {
    rtb_SomaofElements_re

```



```

    += rtDW.RealImagtoComplex.re
    * rtConstP.pooled15[0L];
rtb_SomaofElements_im
    += rtDW.RealImagtoComplex.im
    * rtConstP.pooled15[0L];
j = 1;
}

para (j = 0; j
    < 127; j
    ++) {
rtb_SomaofElements_re
    += rtConstP.pooled15[(int32_T)(1
    + j)]
    * rtDW.RFfilt_states_b[(int32_T)j].re;
rtb_SomaofElements_im
    += rtConstP.pooled15[(int32_T)(1
    + j)]
    * rtDW.RFfilt_states_b[(int32_T)j].im;
}

RFfilt_fz.re = rtb_SomaofElements_re;
RFfilt_fz.im = rtb_SomaofElements_im;

/* Atualizar linha de atraso para o próximo quadro */
para (j = 125; j
    >= 0; j
    --) {
    rtDW.RFfilt_states_b[(int32_T)(1
    + j)] = rtDW.RFfilt_states_b[(int32_T)j];
}

rtDW.RFfilt_states_b[0L] = rtDW.RealImagtoComplex;

/* SwitchCase: '<S30>/Switch Case' incorporados:
 * Ganho: '<S30>/Ganho'
 * Inport: '<S58>/In'
 */
switch ((int32_T)(3.0
    * rtDW.Sonda_op)) {
case 0L:
    /* Saídas paraIfAction SubSystem: '<S30>/y_eq_u' incorporados:
     * ActionPort: '<S58>/Action Port'
     */
    rtDW.Merge_m = RFfilt_fz;

    /* Fim das Saídas paraSubSystem: '<S30>/y_eq_u' */
    break;

case 1L:
    /* Saídas paraIfAction SubSystem: '<S30>/amonly' incorporados:
     * ActionPort: '<S54>/Action Port'
     */
    /* ComplexToMagnitudeAngle: '<S54>/Complex para Magnitude-Angle1'
    incorporados:
     * DiscreteFir: '<S29>/RFfilt'
     */

```

```

rtb_ComplextoMagnitudeAngle1__j = rt_hypotd_snf(rtb_SomaofElements_re,
rtb_SomaofElements_im);
rtb_ComplextoRealImag_o2 = rt_atan2d_snf(rtb_SomaofElements_im,
rtb_SomaofElements_re);

/* Lookup: '<S63>/Look-Up Table'
* Sobre '<S63>/Look-Up Table':
* Input0 Tipo de Dado: Ponto Flutuante real_T
* Output0 Tipo de Dado: Ponto Flutuante real_T
* Lookup Method: Linear_Endpoint
*
* XData está alinhado e uniparamemente espaçado, então o algoritmo só
precisa
* o valor do primeiro elemento, o último elemento e o espaçamento.
* Para eficiência, o XData é excluído do código gerado.
* O parâmetro YData usa o mesmo Tipo de Dado e é dimensionado como Output0
* /
LookupEven_real_T_real_T( &(rtb_LookUpTable_gc), rtConstP.pooled6,
rtb_ComplextoMagnitudeAngle1__j, 1.0, 1U, 1.0);

/* Ganho: '<S54>/Output Ganho' incorporados:
* MagnitudeAngleToComplex: '<S54>/Magnitude-Angle para Complex1'
*/
rtDW.Merge_m.re = rtb_LookUpTable_gc
* cos(rtb_ComplextoRealImag_o2)
* 0.031622776601683791;
rtDW.Merge_m.im = rtb_LookUpTable_gc
* sin(rtb_ComplextoRealImag_o2)
* 0.031622776601683791;

/* Fim das Saídas paraSubSystem: '<S30>/amonly' */
break;

case 2L:
/* Saídas paraIfAction SubSystem: '<S30>/ampm' incorporados:
* ActionPort: '<S55>/Action Port'
*/
/* ComplexToMagnitudeAngle: '<S55>/Complex para Magnitude-Angle1'
incorporados:
* DiscreteFir: '<S29>/RFfilt'
*/
rtb_ComplextoMagnitudeAngle1__j = rt_hypotd_snf(rtb_SomaofElements_re,
rtb_SomaofElements_im);

/* Lookup: '<S64>/Look-Up Table'
* Sobre '<S64>/Look-Up Table':
* Input0 Tipo de Dado: Ponto Flutuante real_T
* Output0 Tipo de Dado: Ponto Flutuante real_T
* Lookup Method: Linear_Endpoint
*
* XData está alinhado e uniparamemente espaçado, então o algoritmo só
precisa
* o valor do primeiro elemento, o último elemento e o espaçamento.
* Para eficiência, o XData é excluído do código gerado.
* O parâmetro YData usa o mesmo Tipo de Dado e é dimensionado como Output0
* /

```

```

LookUpEven_real_T_real_T( &(rtb_LookUpTable_b0), rtConstP.pooled6,
    rtb_ComplextoMagnitudeAngle1__j, 1.0, 1U, 1.0);

/* Soma: '<S55>/Soma1' incorporados:
 * ComplexToMagnitudeAngle: '<S55>/Complex para Magnitude-Angle1'
 * DiscreteFir: '<S29>/Rffilt'
 * Lookup: '<S65>/Look-Up Table'
 */
rtb_ComplextoMagnitudeAngle1__j = rt_Lookup(rtConstP.pooled6, 2,
    rtb_ComplextoMagnitudeAngle1__j, rtConstP.pooled6)
    + rt_atan2d_snf(rtb_SomaofElements_im, rtb_SomaofElements_re);

/* Ganho: '<S55>/Output Ganho' incorporados:
 * MagnitudeAngleToComplex: '<S55>/Magnitude-Angle para Complex1'
 */
rtDW.Merge_m.re = rtb_LookUpTable_b0
    * cos(rtb_ComplextoMagnitudeAngle1__j)
    * 0.031622776601683791;
rtDW.Merge_m.im = rtb_LookUpTable_b0
    * sin(rtb_ComplextoMagnitudeAngle1__j)
    * 0.031622776601683791;

/* Fim das Saídas paraSubSystem: '<S30>/ampm' */
break;

case 3L:
    /* Saídas paraIfAction SubSystem: '<S30>/poly' incorporados:
     * ActionPort: '<S57>/Action Port'
     */
    /* ComplexToMagnitudeAngle: '<S57>/Complex para Magnitude-Angle1'
    incorporados:
     * DiscreteFir: '<S29>/Rffilt'
     */
    rtb_ComplextoMagnitudeAngle1__j = rt_atan2d_snf(rtb_SomaofElements_im,
        rtb_SomaofElements_re);
    rtb_ComplextoRealImag_o2 = rt_hypotd_snf(rtb_SomaofElements_re,
        rtb_SomaofElements_im);

    /* Saturação: '<S71>/Entrada Saturada' */
    if (rtb_ComplextoRealImag_o2 > 408.248290463863) {
        rtb_ComplextoRealImag_o2 = 408.248290463863;
    }

    /* Fim da Saturação: '<S71>/ Entrada Saturada' */

    /* Matemática: '<S71>/Matemática Function' incorporados:
     * Constante: '<S71>/Constante'
     */
    rtb_SomaofElements_re = rt_powd_snf(rtb_ComplextoRealImag_o2, 3.0);

    /* Matemática: '<S71>/Matemática Function1' incorporados:
     * Constante: '<S71>/Constante1'
     */
    rtb_SomaofElements_im = rt_powd_snf(rtb_ComplextoRealImag_o2, 5.0);

    /* Matemática: '<S71>/Matemática Function2' incorporados:
     * Constante: '<S71>/Constante2'

```

```

    */
    rtb_ComplextoMagnitudeAngle1_n1 =
rt_powd_snf(rtb_ComplextoRealImag_o2, 7.0);

/* Ganho: '<S57>/Output Ganho' incorporados:
* Ganho: '<S71>/c1'
* Ganho: '<S71>/c3'
* Ganho: '<S71>/c5'
* Ganho: '<S71>/c7'
* MagnitudeAngleToComplex: '<S57>/Magnitude-Angle para Complex1'
* Soma: '<S71>/Soma'
*/
rtDW.Merge_m.re = (((31.622776601683793
    * rtb_ComplextoRealImag_o2
    + -6.3245553203367591E-5
    * rtb_SomaofElements_re)
    + 0.0
    * rtb_SomaofElements_im)
    + 0.0
    * rtb_ComplextoMagnitudeAngle1_n1)
    * cos(rtb_ComplextoMagnitudeAngle1__j)
    * 0.031622776601683791;
rtDW.Merge_m.im = (((31.622776601683793
    * rtb_ComplextoRealImag_o2
    + -6.3245553203367591E-5
    * rtb_SomaofElements_re)
    + 0.0
    * rtb_SomaofElements_im)
    + 0.0
    * rtb_ComplextoMagnitudeAngle1_n1)
    * sin(rtb_ComplextoMagnitudeAngle1__j)
    * 0.031622776601683791;

/* Fim das Saídas paraSubSystem: '<S30>/poly' */
break;

case 4L:
/* Saídas paraIfAction SubSystem: '<S30>/p2d' incorporados:
* ActionPort: '<S56>/Action Port'
*/
p2d(RFfilt_fz, &rtDW.Merge_m, (ConstB_p2d *)&rtConstB.p2d_p,
&rtDW.p2d_p);

/* Fim das Saídas paraSubSystem: '<S30>/p2d' */
break;
}

/* Fim do SwitchCase: '<S30>/Switch Case' */

/* SwitchCase: '<S53>/Switch Case' incorporados:
* Ganho: '<S53>/Ganho'
*/
if ((int32_T)(0.0
    * rtDW.Sonda_oa)
    == 1L) {
/* Saídas paraIfAction SubSystem: '<S53>/phasenoise' incorporados:
* ActionPort: '<S60>/Action Port'

```

```

    */
    phasenoise(rtDW.Merge_m, &rtb_SineWave, &rtDW.phasenoise_a);

    /* Fim das Saídas paraSubSystem: '<S53>/phasenoise' */
}

/* Fim do SwitchCase: '<S53>/Switch Case' */

/* Update para S-Function (sdspvdly2): '<S1>/Delay Return' */
rtDW.DelayReturn_BUFF[rtDW.DelayReturn_BUFF_OFFSET] = rtDW.Ganho;
rtDW.DelayReturn_BUFF_OFFSET
++;
while (rtDW.DelayReturn_BUFF_OFFSET
    >= 101L) {
    rtDW.DelayReturn_BUFF_OFFSET
    -= 101L;
}

/* Fim do Update para S-Function (sdspvdly2): '<S1>/Atraso de Retorno' */

* Atualizar tempo absoluto * /
/* O "clockTick0" conta o número de vezes que o código desta tarefa foi
* executado. O tempo absoluto é a multiplicação de "clockTick0"
* e "Timing.stepSize0". Tamanho de "clockTick0" garante que o timer não
será
* estouro durante a vida útil da aplicação selecionada.
* O temporizador desta tarefa consiste em dois inteiros não assinados de
32 bits.
* Os dois inteiros representam os bits baixos Timing.clockTick0 e os bits
altos
* Timing.clockTickH0. Quando o bit baixo transborda para 0, os bits altos
aumentam.
*/
if (!(++rtM->Timing.clockTick0)) {
    ++rtM->Timing.clockTickH0;
}

rtM->Timing.t[0] = rtM->Timing.clockTick0 * rtM->Timing.stepSize0 +
    rtM->Timing.clockTickH0 * rtM->Timing.stepSize0 * 4294967296.0;

/* Atualizar tempo absoluto * /
/* O "clockTick1" conta o número de vezes que o código desta tarefa foi
* executado. A resolução deste temporizador inteiro é 1.5625E-5, que é o
tamanho do passo
* da tarefa. Tamanho de "clockTick1" garante que o temporizador não
transbordará durante o
* duração da aplicação selecionada.
* O temporizador desta tarefa consiste em dois inteiros não assinados de
32 bits.
* Os dois inteiros representam os bits baixos Timing.clockTick1 e os bits
altos
* Timing.clockTickH1. Quando o bit baixo transborda para 0, os bits altos
aumentam.
*/
rtM->Timing.clockTick1++;
if (!rtM->Timing.clockTick1) {
    rtM->Timing.clockTickH1++;
}

```

```

}
}

/* Função de Passo do Modelo TID2 */
void ModeloVirtual_step2(void) /* Tempo da amostra: [0.000125s,
0.0s] */
{
    /* (nenhum código de saída / atualização é necessário) */
}

/* Função de Inicialização do Modelo*/
void ModeloVirtual_initialize(void)
{
    /* Código Registrado */

    /* Inicialização não-finitos */
    rt_InitInfAndNaN(sizeof(real_T));

    {
        /* Configurar o objeto solucionador */
        rtsiSetSimTimeStepPtr(&rtM->solverInfo, &rtM->Timing.simTimeStep);
        rtsiSetTPtr(&rtM->solverInfo, &rtmGetTPtr(rtM));
        rtsiSetStepSizePtr(&rtM->solverInfo, &rtM->Timing.stepSize0);
        rtsiSetErrorStatusPtr(&rtM->solverInfo, ((const char_T **)
            (&rtmGetErrorStatus(rtM))));
        rtsiSetRTModelPtr(&rtM->solverInfo, rtM);
    }

    rtsiSetSimTimeStep(&rtM->solverInfo, MAJOR_TIME_STEP);
    rtsiSetSolverName(&rtM->solverInfo, "FixedStepDiscrete");
    rtmSetTPtr(rtM, &rtM->Timing.tArray[0]);
    rtM->Timing.stepSize0 = 1.5625E-5;

    {
        uint32_T RandomSource_SEED_DWORK;
        int16_T i;

        /* Início da Sondagem: '<S116>/Sonda' */
        rtDW.Sonda = 1.0;

        /* Início da Sondagem: '<S93>/Sonda' */
        rtDW.Sonda_k = 1.0;

        /* Início da Sondagem: '<S96>/Sonda' */
        rtDW.Sonda_f = 1.0;

        /* Início da Sondagem: '<S91>/Sonda' */
        rtDW.Sonda_o = 1.0;

        /* Início para Espaço de Trabalho: '<S16>/Espaço de Trabalho' */
        {
            static creal_T pDataValues0[] = { { 1.0, 0.0 }, { 0.99992351138801694,
0.0
}, { 0.99877641621426128, 0.0 }, { 0.99381073849731638, 0.0 }, {
0.98048248711662533, 0.0 }, { 0.95257369832444583, 0.0 }, {
0.90249651907742623, 0.0 }, { 0.82190143897795853, 0.0 }, {
0.70272037124899023, 0.0 }, { 0.53874762957797373, 0.0 }, {

```

0.32777624827017693, 0.0 }, { 0.074142752551646207, 0.0 }, { -
0.20871816632333509, 0.0 }, { -0.49642528951002618, 0.0 }, { -
0.75353628923019544, 0.0 }, { -0.93635898277054919, 0.0 }, { -
0.99969405725308313, 0.0 }, { -0.907754419460904, 0.0 }, { -
0.64803617210160547, 0.0 }, { -0.24485438238350196, 0.0 }, {
0.23284398260064074, 0.0 }, { 0.67584347319081728, 0.0 }, {
0.95626456702012752, 0.0 }, { 0.96645730641436045, 0.0 }, {
0.66667588637627939, 0.0 }, { 0.12336963808359314, 0.0 }, { -
0.48565077338758311, 0.0 }, { -0.91785283987875443, 0.0 }, { -
0.96320690207465709, 0.0 }, { -0.55942074746795323, 0.0 }, {
0.13563387837362442, 0.0 }, { 0.77740535318562565, 0.0 }, {
0.99510865917160662, 0.0 }, { 0.61933674903050862, 0.0 }, { -
0.16009823929579667, 0.0 }, { -0.849066082627079, 0.0 }, { -
0.9487371075487715, 0.0 }, { -0.33943606252240544, 0.0 }, {
0.54912619042306987, 0.0 }, { 0.99931167269645527, 0.0 }, {
0.58978471317052183, 0.0 }, { -0.362598137324665, 0.0 }, { -
0.98504546331726317, 0.0 }, { -0.63856689609321637, 0.0 }, {
0.37409685460479292, 0.0 }, { 0.99625435088567194, 0.0 }, {
0.50712386386977415, 0.0 }, { -0.57975156391760663, 0.0 }, { -
0.97531956790516261, 0.0 }, { -0.14787736976947133, 0.0 }, {
0.88009266285518806, 0.0 }, { 0.72862986607058622, 0.0 }, { -
0.44182642533538435, 0.0 }, { -0.98283915121942422, 0.0 }, { -
0.061802963460090309, 0.0 }, { 0.95980914901696779, 0.0 }, {
0.46388052024208992, 0.0 }, { -0.792725464780417, 0.0 }, { -
0.72010308674960066, 0.0 }, { 0.59972763879526936, 0.0 }, {
0.85553507240851845, 0.0 }, { -0.45288811357192776, 0.0 }, { -
0.91287345409324938, 0.0 }, { 0.38553834358660483, 0.0 }, {
0.922691815084807, 0.0 }, { -0.40824264452879477, 0.0 }, { -
0.89156735953334421, 0.0 }, { 0.51774485982861806, 0.0 }, {
0.80020419848001911, 0.0 }, { -0.69386709422893211, 0.0 }, { -
0.60957881975070627, 0.0 }, { 0.88589777228522582, 0.0 }, {
0.28065351853009757, 0.0 }, { -0.99808836968855186, 0.0 }, {
0.18446463842776051, 0.0 }, { 0.8971005572818268, 0.0 }, { -
0.68490767134698216, 0.0 }, { -0.47480196375801786, 0.0 }, {
0.98900570448813085, 0.0 }, { -0.2207979629916921, 0.0 }, { -
0.81479330147903128, 0.0 }, { 0.86187318480956776, 0.0 }, {
0.086471199490754255, 0.0 }, { -0.92738963945841324, 0.0 }, {
0.76956631891856775, 0.0 }, { 0.17229461743792532, 0.0 }, { -
0.93194559433943824, 0.0 }, { 0.80756051916271931, 0.0 }, {
0.037096911092606245, 0.0 }, { -0.83573944922532384, 0.0 }, {
0.9406291296038426, 0.0 }, { -0.31606629171739181, 0.0 }, { -
0.52828665261608765, 0.0 }, { 0.98710108589464285, 0.0 }, { -
0.78512546213985268, 0.0 }, { 0.098786418305782983, 0.0 }, {
0.628999933893743, 0.0 }, { -0.9907590277344599, 0.0 }, {
0.82888384427637851, 0.0 }, { -0.26876097893061324, 0.0 }, { -
0.39692085398288945, 0.0 }, { 0.87415291929275962, 0.0 }, { -
0.99236078741510381, 0.0 }, { 0.745347746038302, 0.0 }, { -
0.25682732502265776, 0.0 }, { -0.29250312453341665, 0.0 }, {
0.73704518161737176, 0.0 }, { -0.97251410260555327, 0.0 }, {
0.96955986479824052, 0.0 }, { -0.76160955853238521, 0.0 }, {
0.41950198324820914, 0.0 }, { -0.024734427279973537, 0.0 }, { -
0.35104395078808015, 0.0 }, { 0.65740631333535382, 0.0 }, { -
0.86807945024341371, 0.0 }, { 0.97797583152480183, 0.0 }, { -
0.99724763837477459, 0.0 }, { 0.94475538160409922, 0.0 }, { -
0.8424672050733476, 0.0 }, { 0.71146614805742348, 0.0 }, { -
0.56962972587984118, 0.0 }, { 0.43069714771881573, 0.0 }, { -
0.30430798422073191, 0.0 }, { 0.19660644052927662, 0.0 }, { -

```

        0.11108652504880298, 0.0 }, { 0.049453719922709416, 0.0 }, { -
        0.01236815966335196, 0.0 }, { -2.4474366563849412E-15, 0.0 } } ;

    rtDW.FromWorkspace_PWORK.TimePtr = (void *) 0;
    rtDW.FromWorkspace_PWORK.DataPtr = (void *) pDataValues0;
    rtDW.FromWorkspace_IWORK.PrevIndex = -1;
}

/* Início da Sondagem: '<S19>/Sonda' */
/* Modo trigonometrico: compute acumulado
   argumento fcn de trigonometria normalizado para cada canal */
/* Mantenha o valor normalizado no intervalo [0 2 * pi) */
rtDW.Sonda_o1 = 1.0;
rtDW.Sonda_o2[0] = 1.5625E-5;
rtDW.Sonda_o2[1] = 0.0;

/* Início da Sondagem: '<S28>/Sonda' */
rtDW.Sonda_fd = 1.0;

/* Início da Sondagem: '<S33>/Sonda' */
rtDW.Sonda_g = 1.0;

/* Início da Sondagem: '<S30>/Sonda' */
rtDW.Sonda_op = 1.0;

/* Início da Sondagem: '<S53>/Sonda' */
rtDW.Sonda_oa = 1.0;
para (i = 0; i < 127; i++) {
    /* Condições de Inicialização para DiscreteFir: '<S90>/Rffilt' */
    rtDW.Rffilt_states[i].re = 2.2204460492503131E-16;
    rtDW.Rffilt_states[i].im = 0.0;

    /* Condições de Inicialização para DiscreteFir: '<S92>/Rffilt' */
    rtDW.Rffilt_states_l[i].re = 2.2204460492503131E-16;
    rtDW.Rffilt_states_l[i].im = 0.0;

    /* Condições de Inicialização para DiscreteFir: '<S94>/Rffilt' */
    rtDW.Rffilt_states_p[i].re = 2.2204460492503131E-16;
    rtDW.Rffilt_states_p[i].im = 0.0;
}

/* Condições de Inicialização para S-Function (sdsprandsrc2):
'<S135>/Fonte Aleatória' */
RandomSource_SEED_DWORK = 67987UL;
RandSrcInitState_GZ (&RandomSource_SEED_DWORK,
rtDW.RandomSource_STATE_DWORK,
    1L);

/* Condições de Inicialização para S-Function (sdspvdy2): '<S1>/Retorno
de Atraso' */
rtDW.DelayReturn_BUFF_OFFSET = 100L;

/* Condições de Inicialização para S-Function (sdsprandsrc2):
'<S74>/Retorno de Atraso' */
/* Este código só é executado quando o bloco é reativado em um Subsistema
habilitado quando o subsistema habilitado a reativação está definida como
'Reset' */

```



```

    /* Redefinir para o tempo zero em reativar */
    /* Modo trigonometrico: compute acumulado argumento fcn de
    trigonometria normalizado para cada canal */
    /* Mantenha o valor normalizado no intervalo [0 2 * pi) */
    RandSrcInitState_GZ(&RandomSource_SEED_DWORK,
        rtDW.RandomSource_STATE_DWORK_m, 1L);
para (i = 0; i < 127; i++) {
    /* Condições de Inicialização para DiscreteFir: '<S27>/Rffilt' */
    rtDW.Rffilt_states_a[i].re = 2.2204460492503131E-16;
    rtDW.Rffilt_states_a[i].im = 0.0;

    /* Condições de Inicialização para DiscreteFir: '<S29>/Rffilt' */
    rtDW.Rffilt_states_b[i].re = 2.2204460492503131E-16;
    rtDW.Rffilt_states_b[i].im = 0.0;
}

/* SystemInitialize para IfAction SubSystem: '<S96>/phasenoise' */
phasenoise_Init(&rtDW.phasenoise_k);

/* Fim do SystemInitialize para SubSystem: '<S96>/phasenoise' */

/* SystemInitialize para IfAction SubSystem: '<S116>/phasenoise' */
phasenoise_Init(&rtDW.phasenoise_o);

/* Fim do SystemInitialize para SubSystem: '<S116>/phasenoise' */

/* SystemInitialize para IfAction SubSystem: '<S33>/phasenoise' */
/* Condições de Inicialização para S-Function (sdsprandsrc2):
'<S41>/Fonte Aleatória' */
RandomSource_SEED_DWORK = 67988UL;
RandSrcInitState_GZ(&RandomSource_SEED_DWORK,
    rtDW.RandomSource_STATE_DWORK_h, 1L);

/* Fim do SystemInitialize para SubSystem: '<S33>/phasenoise' */

/* SystemInitialize para IfAction SubSystem: '<S53>/phasenoise' */
phasenoise_Init(&rtDW.phasenoise_a);

/* Fim do SystemInitialize para SubSystem: '<S53>/phasenoise' */
}
}

/*
* Arquivo para o código gerado.
*
* [EOF]
*/

```

APÊNDICE C – Código Fonte EE

```

/*
 * Arquivo: CodigoFonteEE.c
 *
 *
 * Versão do Código           : 1.00
 * Autor                     : Rômulo da Costa Delmondes
 * C/C++ Código fonte Gerado em : 04 de Abril 10:23:41 2019
 *
 * Operação da Estação Embarcada
 * Objetivos da Geração do Código:
 *   1. Recepção de dados;
 *   2. Confirmação de Entrega;
 *   3. Transmissão de dados;
 *   4. Log de dados;
 */

#include <SoftwareSerial.h> // Biblioteca comunicação serial
#include "EBYTE.h"         // Biblioteca Radiotransmissor modelo
E32-433T30D

//Bibliotecas Data Logger Shield
#include <SPI.h>           // Comunicação periférica
#include <SD.h>            // Cartão de memória

int Chan;
String receiveMessenger;

struct DADO {
    byte Destination;      // Endereço do dispositivo para enviar a
mensagem
    byte localAddress;    // Endereço deste dispositivo
    byte msgCount;        // Contador de mensagens enviadas
    byte tamMensagem;     // Tamanho da mensagem em bytes
    String outgoing;      // Mensagem de envio
};

int count = 1;           // Contador de mensagens enviadas

//criar um objeto File para manipular um arquivo
File myFile;

//seta o pino 10 de comunicação da shield;
byte chipSelect = 10;

// Encapsulamento da estrutura de empacotamento
DADO MyData;

SoftwareSerial ESerial(2, 3); // Define os pinos Tx e Rx do
radiotransmissor

EBYTE Transceiver(&ESerial, 4, 5, 6); // Seta os pinos Aux, M0 e M1 do
radiotransmissor

// Função Transmissão
void sendMessage(String outgoing)
{

```

```

// Empacotamento dos dados de transmissão
MyData.Destination = 0xFE;
MyData.localAddress = 0xF5;
MyData.msgCount = count;
MyData.tamMensagem = outgoing.length();
MyData.outgoing = outgoing;

// Informações para visualização via terminal serial
Serial.print("\n\n\n\n");
Serial.print("\n\r*****");
Serial.print("\n\rResposta ao Mestre: ");
Serial.print("\n\rDestino: "); Serial.println(MyData.Destination);
Serial.print("\n\rRemetente: "); Serial.println(MyData.localAddress);
Serial.print("\n\rContador: "); Serial.println(MyData.msgCount);
Serial.print("\n\rTamanho da Mensagem: ");
Serial.println(MyData.tamMensagem);
Serial.print("\rMensagem: "); Serial.println(MyData.outgoing);
Serial.print("\n*****");

Serial.print("\n\rAviso de Retransmissão: Pacote nº ");
Serial.println(MyData.msgCount);
Transceiver.SendStruct(&MyData, sizeof(MyData));
MyData.outgoing = "";

count++; // Contador do número de mensagens
enviadas
}

// Função Recepção
void onReceive()
{
// Desempacotamento dos dados de transmissão
Transceiver.GetStruct(&MyData, sizeof(MyData));

// Informações para visualização via terminal serial
Serial.print("\n\n\n\n");
Serial.print("\n\r*****");
Serial.print("\n\rRecebeu do Mestre: ");
Serial.print("\n\rDestino: "); Serial.println(MyData.Destination);
Serial.print("\n\rRemetente: "); Serial.println(MyData.localAddress);
Serial.print("\n\rContador: "); Serial.println(MyData.msgCount);
Serial.print("\n\rTamanho da Mensagem: ");
Serial.println(MyData.tamMensagem);
Serial.print("\rMensagem: "); Serial.println(MyData.outgoing);
Serial.print("\n*****");

Serial.print("\n\rAviso de Recebimento: Pacote nº ");
Serial.println(MyData.msgCount);
}

//Função Log
void gravaSD(int tipo)
{
// /* A biblioteca SD tem um metodo para abrir arquivos e esse arquivo aberto
// será armazenado no objeto myFile, descrito ao inicio desse código. Somente
// um arquivo pode ser aberto por vez, portanto nao se esqueça de fechá-lo
// antes de abrir um novo ou poderá acarretar problemas.

```

```

// */

myFile = SD.open("Log1.txt", FILE_WRITE);

// Se o arquivo foi aberto com sucesso, escreve nele
if (myFile) {
    if (tipo == 1) {
        Serial.print("Escrevendo Mensagem de TX no Log.txt\n");
        // Escreve o texto entre aspas no arquivo a ser gravado no cartão
de memória
        myFile.println("CONFIRMACAO DE ENTREGA DA TRANSMISSAO - ORIGEM
SLAVE / DESTINO MESTRE");
    } else if (tipo == 3) {
        Serial.print("Escrevendo Mensagem de TIMEOUT no Log.txt\n");
        // Escreve o texto entre aspas no arquivo a ser gravado no cartão
de memória
        myFile.println("AGUARDANDO TRASMISSAO - ORIGEM MESTRE /
DESTINO SLAVE");
    }
    else if (tipo == 2) {
        Serial.print("Escrevendo Mensagem de RX no Log.txt\n");
        // Escreve o texto entre aspas no arquivo a ser gravado
no cartão de memória
        myFile.println("CONFIRMACAO DE RECEBIMENTO - ORIGEM
MESTRE / DESTINO SLAVE");
    } else {
        Serial.print("Escrevendo Cabecalho do Log.txt\n");
        // Escreve o texto entre aspas no arquivo a ser gravado
no cartão de memória
        myFile.println("LOG DO SUBSISTEMA DE TELEMETRIA,
RASTREIO E COMANDO - PROJETO LAICANSAT - UNB");
    }

    // fim da escrita, fecha-se o arquivo:
    myFile.close();
}
else {
    // caso ocorra alguma falha, envia mensagem de erro
    Serial.println("Nao foi possivel abrir o arquivo");
}

delay(1000);
}

void setup() {

    // Define as a pinagem de conexão e seu tipo de função para o radiotransmissor
    pinMode(4, OUTPUT);
    pinMode(5, OUTPUT);
    pinMode(6, INPUT);

    //inicialização da comunicação serial
    Serial.begin(9600);
    //inicialização do comunicação com o radiotransmissor
    ESerial.begin(9600);

    Transceiver.init();
}

```

```

Transceiver.PrintParameters();

//inicialização do cartão de memória
Serial.print("Iniciando cartão SD...");

if (!SD.begin(chipSelect)) {
  Serial.print("Erro no cartão ou cartão não inserido.");
  return;
}
Serial.print("Cartão inicializado");
}

void loop() {

  int sto;

  //analisa um pacote e chama a função recepção com o resultado:
  if (ESerial.available()) {

    onReceive();

    sto = 1; // Status de Recebimento
    gravaSD(sto); // Função Log

    receiveMessenger = "";
    receiveMessenger = MyData.outgoing;

    count = MyData.msgCount; // Contador do pacote
recebido
  }
  else {
    if (Serial.available()) {
      Chan = Serial.read();

      if (Chan > 47) {
        Serial.println(Chan - 48);
        Transceiver.SetChannel(Chan - 48);
        Transceiver.SaveParameters(PERMANENT);
        Transceiver.PrintParameters();
      }
    }
    delay(1000);

    if (MyData.msgCount == count) {

      //Envia a mensagem
      String mensagem = receiveMessenger;

      sendMessage(mensagem); // Função Transmissão

      sto = 2; // Status de Transmissão
      gravaSD(sto); // Função Log
    }

    Serial.println("\n\rPesquisando: ");
  }
}

```

```
sto = 3; // Status de Aguardando
gravaSD(sto); // Função Log

count++;
}
}
```

APÊNDICE D – Código Fonte ET

```

/*
 * Arquivo: CodigoFonteET.c
 *
 *
 * Versão do Código           : 1.00
 * Autor                     : Rômulo da Costa Delmondes
 * C/C++ Código fonte Gerado em : 04 de Abril 11:07:23 2019
 *
 * Operação da Estação Terrestre
 * Objetivos da Geração do Código:
 *   1. Transmissão de dados;
 *   2. Recepção de dados;
 *   3. Timeout;
 *   4. Display LCD;
 */

#include <SoftwareSerial.h> // Biblioteca comunicação serial
#include "EBYTE.h"         // Biblioteca Radiotransmissor modelo
                             E32-433T30D

// Biblioteca Display LCD
#include <Wire.h>           // Comunicação I2C
#include <LiquidCrystal_I2C.h> // Controle do Display LCD, via
                             comunicação I2C

// Inicializa o display LCD no endereço 0x27
LiquidCrystal_I2C lcd(0x27,2,1,0,4,5,6,7,3, POSITIVE);

// Define as constantes para interface dos pinos de conexão do radiotransmissor
#define PIN_M0 4
#define PIN_M1 5
#define PIN_AX 6

int Chan;
unsigned char seloTimeout;

// Declaração da estrutura do pacote
struct DADO {
    byte Destination;           // Endereço do dispositivo para enviar a
    mensagem
    byte localAddress;         // Endereço deste dispositivo
    byte msgCount;             // Contador de mensagens enviadas
    byte tamMensagem;         // Tamanho da mensagem em bytes
    String outgoing;           // Mensagem de envio
};

long lastSendTime = 0;        // TimeStamp da última mensagem enviada
int interval = 5000;          // Intervalo em ms no envio das mensagens
                               (inicial 5s)
int count = 0;                // Contador de mensagens enviadas

// Encapsulamento da estrutura de empacotamento
DADO MyData;

SoftwareSerial ESerial(2, 3); // Define os pinos Tx e Rx do

```

```

radiotransmissor

EByte Transceiver(&ESerial, 4, 5, 6);    // Seta os pinos Aux, M0 e M1 do
radiotransmissor

void setup() {

    seloTimeOut = 1;

    //inicialização da comunicação serial
    Serial.begin(9600);
    //inicialização do lcd
    lcd.begin (16,2);

    while (!Serial);

    Serial.println(" Comunicacao LoRa Half-Duplex - P2P ");

    ESerial.begin(9600);

    // Inicializa o radio LoRa em 9600bps e checa se esta ok!
    if (!Serial)
    {
        Serial.println(" Erro ao iniciar modulo LoRa. Verifique a conexão dos seus
pinos!! ");
        while (true);
    }
    Serial.println(" Modulo LoRa iniciado com sucesso!!!");

    lcd.setCursor(0,0);
    lcd.print("COM Half Duplex");
    lcd.setCursor(0,1);
    lcd.print("Modulo Iniciado!");

    Transceiver.init();

}

// Loop do microcontrolador - Operações de comunicação LoRa
void loop() {
    // verifica se existe o intervalo de tempo para enviar uma mensagem
    if (millis() - lastSendTime > interval)
    {
        //A variável string
        String mensagem = "UNB";           // Definição da mensagem
        sendMessage(mensagem);

        String status = "tx";              // Status de transmissão
        onLCD(status);                      // Função Display LCD

        lastSendTime = millis();           // Timestamp da ultima mensagem

        if(seloTimeOut==0){
            seloTimeOut = 1;
        }
    }
}

```



```

        status = "tx";           // Status de Transmissão
        onLCD(status);          // Função Display LCD
    }
    // Condição TimeOut
    else {
        Serial.print("\n\rAviso de TimeOut: Pacote n° ");
        Serial.println(MyData.msgCount);

        status = "to";         // Status de TimeOut
        onLCD(status);         // Função LCD
    }
}

//analisa um pacote e chama a função recepção com o resultado:
if (ESerial.available()) {
    count++;
    seloTimeOut = 0;
    onReceive();
    String status = "rx";     // Status de Recebimento
    onLCD(status);           // Função Display LCD
}
}

// Função Transmissão
void sendMessage(String outgoing)
{
    // Empacotamento dos dados de transmissão
    MyData.Destination = 0xF5;
    MyData.localAddress = 0xFE;
    MyData.msgCount = count;
    MyData.tamMensagem = outgoing.length();
    MyData.outgoing = outgoing;

    // Informações para visualização via terminal serial
    Serial.print("\n\n\n");
    Serial.print("\n*****");
    Serial.print("\n\rEnvio para o Escravo: ");
    Serial.print("\n\rDestino: "); Serial.println(MyData.Destination);
    Serial.print("\rRemetente: "); Serial.println(MyData.localAddress);
    Serial.print("\rContador: "); Serial.println(MyData.msgCount);
    Serial.print("\rTamanho da Mensagem: ");
    Serial.println(MyData.tamMensagem);
    Serial.print("\rMensagem enviada: "); Serial.println(MyData.outgoing);
    Serial.print("\n\r*****");

    Serial.print("\n\rAviso de Transmissão: Pacote n° ");
    Serial.println(MyData.msgCount);
    Transceiver.SendStruct(&MyData, sizeof(MyData));
}

// Função Recepção
void onReceive()
{

```

```

// Desempacotamento dos dados de transmissão
Transceiver.GetStruct(&MyData, sizeof(MyData));

// Informações para visualização via terminal serial
Serial.print("\n\n\n");
Serial.print("\n\r*****");
Serial.print("\n\rRecebeu do Escravo: ");
Serial.print("\n\rDestino_Escravo: ");
Serial.println(MyData.Destination);
Serial.print("\rRemetente_Escravo: ");
Serial.println(MyData.localAddress);
Serial.print("\rContador_Escravo: "); Serial.println(MyData.msgCount);
Serial.print("\rTamanho da Mensagem_Escravo: ");
Serial.println(MyData.tamMensagem);
Serial.print("\rMensagem_Escravo: "); Serial.println(MyData.outgoing);
Serial.print("\n\r*****");

Serial.print("\n\rAviso de Entrega: Pacote n° ");
Serial.println(MyData.msgCount);

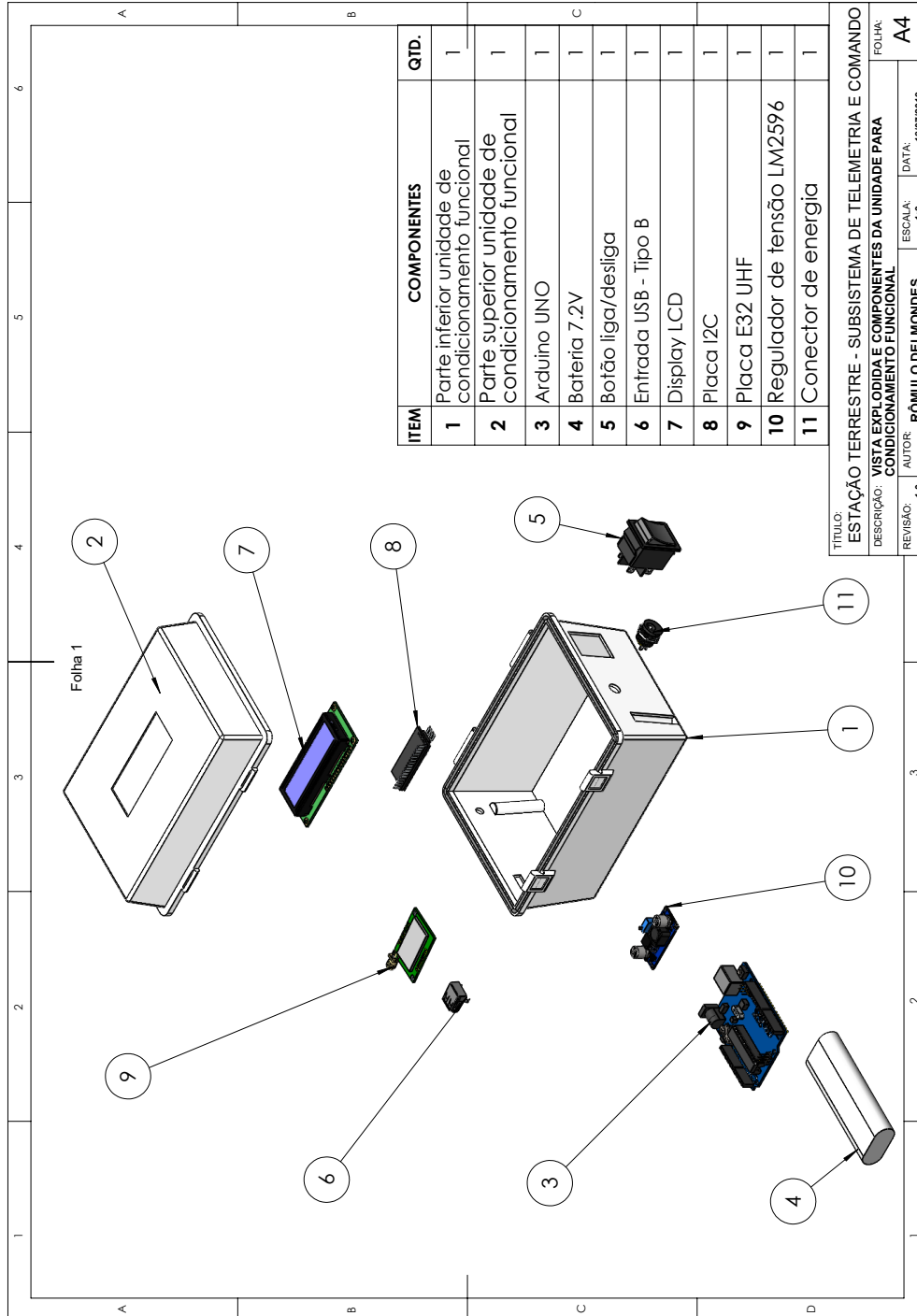
}

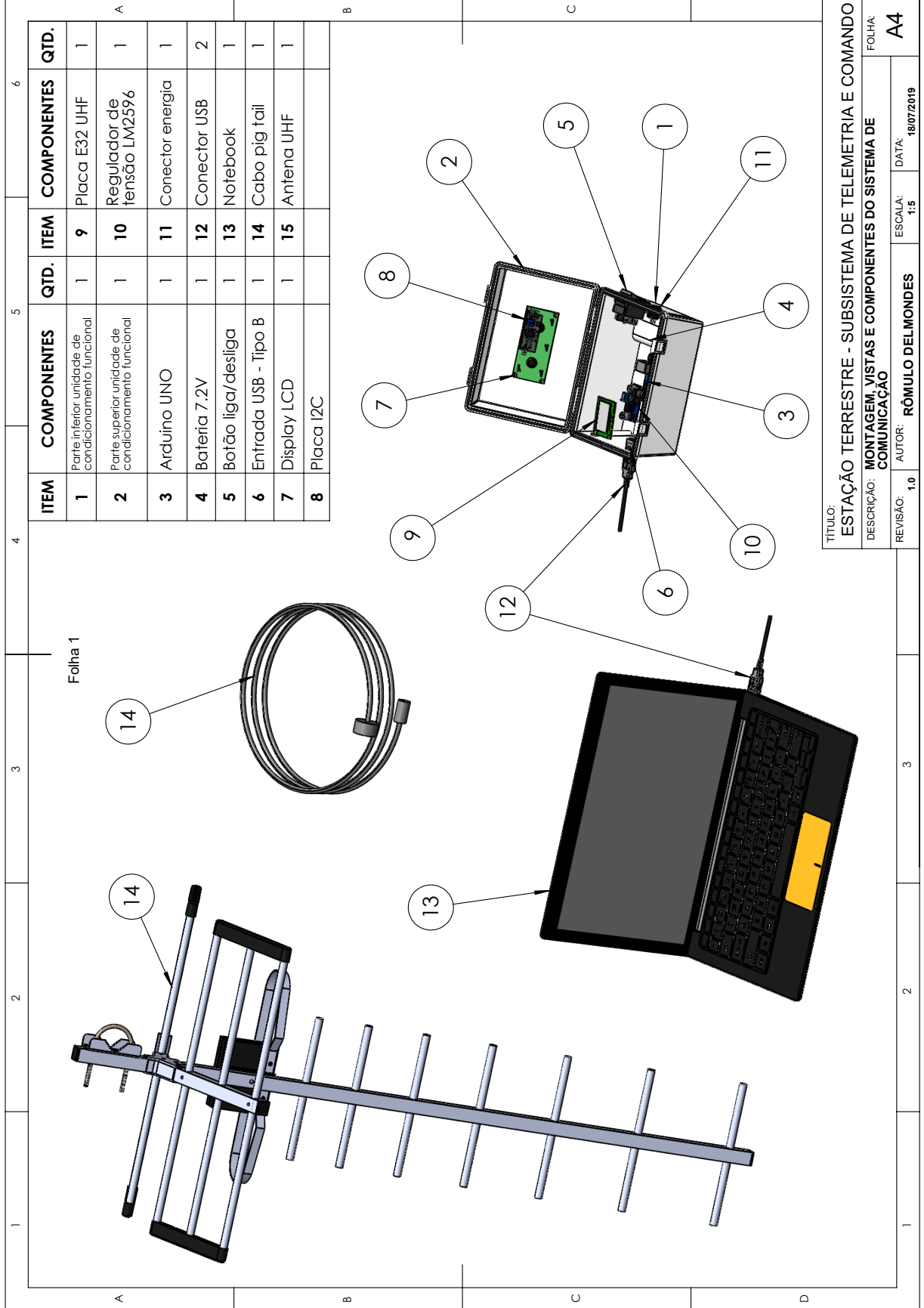
//Função Display LCD
void onLCD(String status)
{
  //Limpa a tela
  lcd.clear();

  if (status == "tx") {
    lcd.setCursor(0, 0);
    //Envia o texto entre aspas para o LCD
    lcd.print("TX - PCT "); lcd.print(MyData.msgCount);
    lcd.setCursor(0, 1);
    lcd.print("MENSAGEM: "); lcd.print(MyData.outgoing);
  } else if (status == "to") {
    lcd.setCursor(0, 0);
    //Envia o texto entre aspas para o LCD
    lcd.print("AVISO TIMEOUT");
    lcd.setCursor(0, 1);
    lcd.print("RTX - PCT "); lcd.print(MyData.msgCount);
  }
  else if (status == "rx") {
    lcd.setCursor(0, 0);
    //Envia o texto entre aspas para o LCD
    lcd.print("RX - PCT "); lcd.print(MyData.msgCount);
    lcd.setCursor(0, 1);
    lcd.print("CONFIRMA ENTREGA");
  }
  delay(3000);
}
}

```

APÊNDICE E – Unidade de Condicionamento Funcional (UCF-ET)





Folha 1

ITEM	COMPONENTES	QTD.	ITEM	COMPONENTES	QTD.
1	Parte inferior unidade de condicionamento funcional	1	9	Placa E32 UHF	1
2	Parte superior unidade de condicionamento funcional	1	10	Regulador de tensão LM2596	1
3	Arduino UNO	1	11	Conector energia	1
4	Bateria 7.2V	1	12	Conector USB	2
5	Botão liga/desliga	1	13	Notebook	1
6	Entrada USB - Tipo B	1	14	Cabo pig tail	1
7	Display LCD	1	15	Antena UHF	1
8	Placa I2C				

TÍTULO: ESTAÇÃO TERRESTRE - SUBSISTEMA DE TELEMETRIA E COMANDO
 DESCRIÇÃO: MONTAGEM, VISTAS E COMPONENTES DO SISTEMA DE COMUNICAÇÃO
 REVISÃO: 1.0 AUTOR: RÔMULO DEL MONDES ESCALA: 1:5 DATA: 18/07/2019
 FOLHA: A4

Anexos

ANEXO A – Resolução nº 697



20

ISSN 1677-7042

Diário Oficial da União - Seção 1

Nº 168, quinta-feira, 30 de agosto de 2018

Santo Antônio da Barra	33 a 35	33 a 01	33 a 02
Santo Antônio de Goiás	33 a 35	33 a 01	33 a 02
Santo Antônio do Descoberto	30 a 36	30 a 01	30 a 02
São Domingos	30 a 34	30 a 36	30 a 01
São Francisco de Goiás	30 a 36	30 a 01	30 a 02
São João d'Alcântara	30 a 35	30 a 35	30 a 01
São João da Paraíma	33 a 35	33 a 01	33 a 01
São Luís de Montes Belos	30 a 35	30 a 01	30 a 01
São Luiz do Norte	34 a 36	34 a 01	34 a 02
São Miguel do Araguaia	30	30	30
São Miguel do Passa Quatro	31 a 35	31 a 01	31 a 02
São Patrício	30 a 36	30 a 01	30 a 02
São Simão	30 a 34	30 a 01	30 a 01
Senador Camedo	31 a 01	31 a 02	31 a 02
Serranópolis	34 a 01	34 a 02	34 a 02
Silvânia	31 a 36	31 a 02	31 a 02
Simolândia	30 a 34	30 a 36	30 a 36
Sítio d'Abadia	30 a 34	30 a 35	30 a 01
Taquaral de Goiás	30 a 35	30 a 01	30 a 02
Teresina de Goiás	30 a 35	30 a 01	30 a 01
Terezópolis de Goiás	30 a 36	30 a 02	30 a 02
Três Ranchos	30 a 34	30 a 01	30 a 01
Trindade	33 a 35	33 a 01	33 a 02
Trombas	30 a 35	30 a 01	30 a 01
Turvânia	30 a 36	30 a 01	30 a 01
Turvelândia	33 a 35	33 a 01	33 a 01
Uirapuru	30 a 35	30 a 01	30 a 01
Uruaçu	34 a 35	34 a 01	34 a 02
Uruana	30 a 36	30 a 01	30 a 02
Uratú	31 a 35	31 a 36	31 a 01
Valparaíso de Goiás	30 a 35	30 a 01	30 a 02
Varijão	30 a 36	30 a 02	30 a 02
Vianópolis	31 a 35	31 a 01	31 a 02
Vicentinópolis	33 a 35	33 a 01	33 a 02
Vila Boa	30 a 35	30 a 01	30 a 02
Vila Propício	31 a 35	31 a 35	31 a 02

Ministério da Ciência, Tecnologia, Inovações e Comunicações

GABINETE DO MINISTRO

PORTARIA Nº 4.290-SEI, DE 24 DE AGOSTO DE 2018

O MINISTRO DE ESTADO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES, no uso da atribuição que lhe confere o art. 87, parágrafo único, inciso IV, da Constituição Federal, em conformidade com o disposto no art. 5º da Lei n.º 5.785, de 23 de junho de 1972, tendo em vista o disposto na Lei n.º 13.424, de 28 de março de 2017 e o disposto no Decreto n.º 52.795, de 31 de outubro de 1963, com nova redação dada pelo Decreto n.º 9.138, de 22 de agosto de 2017 e o consta do Processo Administrativo nº 53000.060582/2013-72, invocando as razões presente na Nota Técnica n.º 18.046/2018/SEI-MCTIC, chancelada pelo Parecer Jurídico n.º00881/2018/CONJUR-MCTIC/CGU/AGU, da Consultoria Jurídica atuante neste MCTIC, resolve:

Art. 1º Renovar, de acordo com o art. 33, § 3º, da Lei no 4.117, de 27 de agosto de 1962, por dez anos, a partir de 22 de janeiro de 2014, a permissão outorgada à Radio Rio Verde Ltda., nos termos da Portaria n.º 344, de 19 de março de 2002, publicada no Diário Oficial da União de 25 de março de 2002, chancelada pelo Decreto Legislativo nº 591, de 2003, publicado no Diário Oficial da União de 27 de agosto de 2003, para executar, sem direito de exclusividade, o serviço de radiodifusão sonora em frequência modulada, no município de Baependi, estado de Minas Gerais.

Art. 2º A execução do serviço de radiodifusão, cuja permissão é renovada por esta Portaria reger-se-á pelo Código Brasileiro de Telecomunicações, leis subsequentes e seus regulamentos.

Art. 3º Este ato somente produzirá efeitos legais após deliberação do Congresso Nacional, nos termos do § 3º do art. 223 da Constituição Federal.

Art. 4º Esta Portaria entra em vigor na data de sua publicação.

GILBERTO KASSAB

GILBERTO KASSAB

PORTARIA Nº 4.412, DE 28 DE AGOSTO DE 2018

Homologa, de forma escalonada, o encerramento da transmissão da programação das emissoras dos serviços de radiodifusão de sons e imagens e de retransmissão de televisão, em tecnologia analógica, dos agrupamentos de municípios de Juazeiro do Norte/CE e Sobral/CE.

O MINISTRO DE ESTADO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES, no uso das atribuições que lhe confere o art. 87, parágrafo único, incisos II e IV da Constituição Federal, e

CONSIDERANDO o disposto no art. 6º, inciso III, da Lei nº 13.341, de 29 de setembro de 2016, que transfere as competências do extinto Ministério das Comunicações para o Ministério da Ciência, Tecnologia, Inovações e Comunicações;

CONSIDERANDO o Decreto nº 5.820, de 29 de junho de 2006, alterado pelos Decretos nº 7.670, de 16 de janeiro de 2012, nº 8.061, de 29 de julho de 2013 e nº 8.753, de 10 de maio de 2016, que dispõe sobre a implantação do Sistema Brasileiro de Televisão Digital Terrestre SBTVD-T e estabelece diretrizes para a transição do sistema de transmissão analógica para o sistema de transmissão digital do Serviço de Radiodifusão de Sons e Imagens (TV) e do Serviço de Retransmissão de Televisão (RTV), e dá outras providências;

CONSIDERANDO o disposto no art. 10 do Decreto nº 5.820, de 29 de junho de 2006, e alterações, segundo o qual o Ministério da Ciência, Tecnologia, Inovações e Comunicações estabelecerá cronograma de transição da transmissão analógica dos serviços de radiodifusão de sons e imagens e de retransmissão de televisão para o Sistema Brasileiro de Televisão Digital Terrestre - SBTVD-T;

CONSIDERANDO o disposto no art. 14 do Decreto nº 5.820, de 29 de junho de 2006, e alterações, segundo o qual o Ministério da Ciência, Tecnologia, Inovações e Comunicações expedirá normas complementares necessárias à execução e operacionalização do SBTVD-T;

CONSIDERANDO o cronograma de transição da transmissão analógica dos serviços TV e RTV para o SBTVD-T, definido pela Portaria MCTIC nº 2.992, de 26 de maio de 2017, que foi alterada pela Portaria MCTIC nº 7.432, de 20 de dezembro de 2017, pela Portaria MCTIC nº 1.019, de 26 de fevereiro de 2018 e pela Portaria MCTIC nº 3.291 de 25 de junho de 2018;

CONSIDERANDO o disposto no art. 4º da Portaria MCTIC nº 2.992, de 26 de maio de 2017, que estabelece como condição para o desligamento da transmissão analógica dos serviços de TV e RTV, que pelo menos 93% (noventa e três por cento) dos domicílios do município que acessem o serviço livre, aberto e gratuito por transmissão terrestre, estejam aptos à recepção da televisão digital terrestre;

CONSIDERANDO o disposto no inciso IV do art. 5º da Portaria MCTIC nº 2.992, de 26 de maio de 2017, que estabelece que cabe ao Grupo de Implantação do Processo de Redistribuição e Digitalização de Canais de TV e RTV - GIREDD, aferir o atingimento do mencionado percentual de domicílios aptos à recepção da televisão digital terrestre;

CONSIDERANDO a decisão tomada na 14ª Reunião Ordinária do GIREDD, de considerar o percentual mínimo para atingimento da condição do desligamento como sendo o de 90 (noventa) pontos percentuais, tendo em vista a margem de erro de 3 (três) pontos percentuais;

CONSIDERANDO o disposto no art. 2º da Portaria MC nº 6.738, de 21 de dezembro de 2015, que estabelece que a concessão de outorgas para a exploração do Serviço de RTV em caráter secundário, com a utilização de tecnologia digital, ocorrerá até a data do desligamento do sinal analógico na localidade, conforme cronograma estabelecido pelo Ministério da Ciência, Tecnologia, Inovações e Comunicações; e

CONSIDERANDO que o GIREDD, em sua 13ª Reunião Extraordinária, realizada em 27 de agosto de 2018, deliberou no sentido de recomendar ao Ministério da Ciência, Tecnologia, Inovações e Comunicações o desligamento escalonado da transmissão analógica dos serviços de TV e RTV, dos agrupamentos de municípios de Juazeiro do Norte/CE e Sobral/CE, conforme disposto no Ofício nº 449/2018/SEI/GPR-ANATEL, encaminhado pelo Presidente do GIREDD, resolve:

Art. 1º Homologar o encerramento da transmissão da programação das emissoras dos serviços de radiodifusão de sons e imagens e de retransmissão de televisão, em tecnologia analógica, com início às 23 horas e 59 minutos do dia 28 de agosto de 2018 e término às 23 horas e 59 minutos do dia 31 de outubro de 2018, dos agrupamentos de municípios de Juazeiro do Norte/CE e Sobral/CE, que abrangem os seguintes municípios do Estado do Ceará: Barbalha, Caririáçu, Crato, Forquilha, Juazeiro do Norte, Massapé, Missão Velha, Santana do Acaraú e Sobral.

Art. 2º Após o início do encerramento da transmissão da programação das emissoras dos serviços de radiodifusão de sons e imagens e de retransmissão de televisão, em tecnologia analógica, não serão concedidas autorizações para exploração do Serviço de RTV em caráter secundário, conforme estabelece o art. 2º da Portaria MC nº 6.738, de 21 de dezembro de 2015.

Art. 3º Esta Portaria entra em vigor na data de sua publicação.

AGÊNCIA NACIONAL DE TELECOMUNICAÇÕES CONSELHO DIRETOR

RESOLUÇÃO Nº 697, DE 28 DE AGOSTO DE 2018

Atribui e destina faixas de radiofrequência ao Serviço de Radioamador e aprova o Regulamento sobre Condições de Uso de Radiofrequências pelo Serviço de Radioamador.

O CONSELHO DIRETOR DA AGÊNCIA NACIONAL DE TELECOMUNICAÇÕES, no uso das atribuições que lhe foram conferidas pelo art. 22 da Lei nº 9.472, de 16 de julho de 1997, e pelos arts. 17 e 35 do Regulamento da Agência Nacional de Telecomunicações, aprovado pelo Decreto nº 2.338, de 7 de outubro de 1997,

CONSIDERANDO a competência da Anatel de adotar as medidas necessárias para o atendimento do interesse público, de acordo com o disposto no art. 19 da Lei nº 9.472, de 1997;

CONSIDERANDO a competência da Anatel de regular o uso eficiente e adequado do espectro, consoante o interesse público, de acordo com o disposto no art. 160 da Lei nº 9.472, de 1997;

CONSIDERANDO os termos dos arts. 159 e 161 da Lei nº 9.472, de 1997, segundo os quais, na destinação de faixas de radiofrequência, será considerado o emprego racional e econômico do espectro e que, a qualquer tempo, poderá ser modificada, desde que o interesse público ou o cumprimento de convenções ou tratados internacionais assim o determine;

CONSIDERANDO o disposto no Regulamento de Radiocomunicações da União Internacional de Telecomunicações - UIT, edição 2016, no qual constam as atribuições ao Serviço de Radioamador aprovadas na Conferência Mundial de Radiocomunicações de 2015 e anteriores;

CONSIDERANDO o benefício para os radioamadores brasileiros em viabilizar a rádio experimentação e a operação em faixas de radiofrequência padronizadas internacionalmente;

CONSIDERANDO as contribuições recebidas em decorrência da Consulta Pública nº 14, de 7 de junho de 2017, publicada no Diário Oficial da União do dia 8 de junho de 2017;

CONSIDERANDO deliberação tomada em sua Reunião nº 857, de 23 de agosto de 2018;

CONSIDERANDO o constante dos autos do Processo nº 53500.026094/2016-48, resolve:

Art. 1º Atribuir e destinar adicionalmente ao Serviço de Radioamador, em caráter primário e sem exclusividade, as faixas de radiofrequência de 1850 kHz a 2000 kHz e de 3800 kHz a 4000 kHz.

Art. 2º Atribuir e destinar adicionalmente ao Serviço de Radioamador, em caráter secundário, as seguintes faixas de radiofrequência:

I - 135,7 kHz a 137,8 kHz, 472 kHz a 479 kHz e 10100 kHz a 10138 kHz, adotando as Notas Internacionais 5.67A e 5.80A; e,

II - 5351,5 kHz a 5366,5 kHz.

Art. 3º Destinar ao Serviço de Radioamador, em caráter secundário, a faixa de 122,25 GHz a 123 GHz.

Art. 4º Revogar a atribuição e destinação da faixa de radiofrequência de 3500 MHz a 3600 MHz ao Serviço de Radioamador.

Art. 5º Manter a destinação das faixas de radiofrequência listadas a seguir ao Serviço de Radioamador, em caráter primário e de forma não exclusiva:

I - 1800 - 2000 kHz;
II - 3500 - 4000 kHz;
III - 7000 - 7100 kHz;
IV - 7100 - 7300 kHz;
V - 14000 - 14250 kHz;
VI - 14250 - 14350 kHz;
VII - 18068 - 18168 kHz;
VIII - 21000 - 21450 kHz;
IX - 24890 - 24990 kHz;
X - 28000 - 29700 kHz;
XI - 50 - 54 MHz;
XII - 144 - 146 MHz;
XIII - 146 - 148 MHz;
XIV - 220 - 225 MHz;
XV - 24 - 24,05 GHz;
XVI - 47 - 47,2 GHz;
XVII - 77,5 - 78 GHz;
XVIII - 134 - 136 GHz; e,
XIX - 248 - 250 GHz.

Parágrafo único. As faixas de radiofrequência dispostas nos incisos III, V, VII, VIII, IX, X, XII, XV, XVI, XVII, XVIII e XIX do caput deste artigo poderão ser utilizadas também para aplicações de radioamador por satélite, respeitado o caráter da faixa.

Art. 6º Manter a destinação das faixas de radiofrequência listadas a seguir ao Serviço de Radioamador, em caráter secundário e de forma não exclusiva:

I - 135,7 kHz - 137,8 kHz;
II - 472 - 479 kHz;
III - 5351,5 - 5366,5 kHz;
IV - 10100 - 10150 kHz;
V - 430 - 435 MHz;
VI - 435 - 438 MHz;
VII - 438 - 440 MHz;
VIII - 902 - 907,5 MHz;
IX - 915 - 928 MHz;
X - 1240 - 1260 MHz;



- XI - 1260 - 1270 MHz;
- XII - 1270 - 1300 MHz;
- XIII - 2300 - 2400 MHz;
- XIV - 2400 - 2450 MHz;
- XV - 3300 - 3400 MHz;
- XVI - 3400 - 3410 MHz;
- XVII - 3410 - 3500 MHz;
- XVIII - 5650 - 5670 MHz;
- XIX - 5670 - 5830 MHz;
- XX - 5830 - 5850 MHz;
- XXI - 5850 - 5925 MHz;
- XXII - 10 - 10,45 GHz;
- XXIII - 10,45 - 10,50 GHz;
- XXIV - 24,05 - 24,25 GHz;
- XXV - 76 - 77,5 GHz;
- XXVI - 78 - 81 GHz;
- XXVII - 122,25 - 123 GHz;
- XXVIII - 136 - 141 GHz; e,
- XXIX - 241 - 248 GHz.

§ 1º As faixas de radiofrequência dispostas nos incisos XX, XXIII, XXV, XXVI, XXVIII e XXIX do caput deste artigo poderão ser utilizadas também para aplicações de radioamador por satélite, respeitando o caráter da faixa.

§ 2º As faixas de radiofrequência dispostas nos incisos VI, XI, XIV, XVI e XVIII do caput deste artigo poderão ser utilizadas também para aplicações de radioamador por satélite, devendo-se observar o disposto na Nota Internacional 5.282 do Regulamento de Radiocomunicações da UIT (RR).

Art. 7º Aprovar, na forma do Anexo, o Regulamento sobre Condições de Uso de Radiofrequências pelo Serviço de Radioamador.

Art. 8º Revogar a Resolução nº 452, de 11 de dezembro de 2006, publicada no Diário Oficial da União de 14 de dezembro de 2006.

Art. 9º Esta Resolução entra em vigor no prazo de 90 (noventa) dias contados da data da sua publicação.

JUAREZ MARTINHO QUADROS DO NASCIMENTO
Presidente do Conselho

ANEXO

REGULAMENTO SOBRE CONDIÇÕES DE USO DE RADIOFREQUÊNCIAS PELO SERVIÇO DE RADIOAMADOR

CAPÍTULO I

DOS OBJETIVOS

Art. 1º Este Regulamento tem por objetivo estabelecer as condições de uso de radiofrequência pelo Serviço de Radioamador.

CAPÍTULO II
DAS FAIXAS DE RADIOFREQUÊNCIA E DAS CONDIÇÕES DE USO

Art. 2º As estações do Serviço de Radioamador devem ser operadas de acordo com a Classe do Certificado de Operador de Estação de Radioamador (COER) do radioamador que a utiliza, definida no Regulamento do Serviço de Radioamador.

Art. 3º A denominação das faixas de radiofrequência e as Classes do COER autorizadas a operar em cada uma delas estão definidas na tabela a seguir.

Tabela
Denominação das faixas de radiofrequência e classes do COER autorizadas do Serviço de Radioamador

Denominação baseada no comprimento de onda	Faixas de Radiofrequência	Caráter de Utilização	Classes do COER autorizadas
Faixa de 2200 metros	135,7 - 137,8 kHz	Secundário	A
Faixa de 630 metros	472 - 479 kHz	Secundário	A
Faixa de 160 metros	1800 - 1850 kHz	Primário	Todas as classes
	1850 - 2000 kHz	Primário	A
Faixa de 80 metros	3500 - 3800 kHz	Primário	Todas as classes
	3800 - 4000 kHz	Primário	A
Faixa de 60 metros	5351,5 - 5366,5 kHz	Secundário	A
Faixa de 40 metros	7000 - 7047 kHz	Primário	Todas as classes
	7047 - 7300 kHz	Primário	A e B
Faixa de 30 metros	10100 - 10150 kHz	Secundário	A
Faixa de 20 metros	14000 - 14350 kHz	Primário	A
Faixa de 17 metros	18068 - 18168 kHz	Primário	A
Faixa de 15 metros	21000 - 21150 kHz	Primário	Todas as classes
	21150 - 21300 kHz	Primário	A e B
	21300 - 21450 kHz	Primário	A
Faixa de 12 metros	24890 - 24990 kHz	Primário	Todas as classes
Faixa de 10 metros	28000 - 29700 kHz	Primário	Todas as classes
Faixa de 6 metros	50 - 54 MHz	Primário	Todas as classes
Faixa de 2 metros	144 - 148 MHz	Primário	Todas as classes
Faixa de 1,3 metro	220 - 225 MHz	Primário	Todas as classes
Faixa de 70 centímetros	430 - 440 MHz	Secundário	Todas as classes
Faixa de 33 centímetros	902 - 907,5 MHz	Secundário	Todas as classes
	915 - 928 MHz	Secundário	
Faixa de 23 centímetros	1240 - 1300 MHz	Secundário	Todas as classes
Faixa de 13 centímetros	2300 - 2450 MHz	Secundário	Todas as classes
Faixa de 9 centímetros	3300 - 3500 MHz	Secundário	Todas as classes
Faixa de 5 centímetros	5650 - 5925 MHz	Secundário	Todas as classes

Faixa de 3 centímetros	10 - 10,50 GHz	Secundário	Todas as classes
Faixa de 1,2 centímetro	24 - 24,05 GHz	Primário	A
	24,05 - 24,25 GHz	Secundário	A
Faixa de 6 milímetros	47 - 47,2 GHz	Primário	A
Faixa de 4 milímetros	76 - 77,5 GHz	Secundário	A
	77,5 - 78 GHz	Primário	A
	78 - 81 GHz	Secundário	A
Faixa de 2,5 milímetros	122,25 - 123 GHz	Secundário	A
Faixa de 2 milímetros	134 - 136 GHz	Primário	A
	136 - 141 GHz	Secundário	A
Faixa de 1 milímetro	241 - 248 GHz	Secundário	A
	248 - 250 GHz	Primário	A

Art. 4º Ressalvadas as condições específicas previstas neste Regulamento, os limites gerais de potência por Classe do COER são os seguintes:

I - a potência média na saída do transmissor de uma estação do Serviço de Radioamador, quando operada por Radioamador Classe A, deve estar limitada a 1.500 watts na fração de tempo em que o sistema permanece ativo (duty cycle);

II - a potência média na saída do transmissor de uma estação do Serviço de Radioamador, quando operada por Radioamador Classe B, deve estar limitada a 1.000 watts na fração de tempo em que o sistema permanece ativo (duty cycle); e,

III - a potência média na saída do transmissor de uma estação do Serviço de Radioamador, quando operada por Radioamador Classe C, deve estar limitada a 100 watts na fração de tempo em que o sistema permanece ativo (duty cycle).

Parágrafo único. Ato da Superintendência responsável pela administração do uso do espectro de radiofrequências poderá definir limites diferenciados para determinadas faixas de radiofrequência devido a condições específicas, incluindo a convivência com outros serviços de radiocomunicações.

Art. 5º Para as estações do Serviço de Radioamador operando nas faixas de radiofrequência relacionadas neste artigo, prevalecem os seguintes limites de potência:

I - de 135,7 kHz a 137,8 kHz, o limite não pode exceder a 1 watt (e.i.r.p.);

II - de 472 kHz a 479 kHz, o limite não pode exceder a 5 watts (e.i.r.p.) e não podem causar interferência prejudicial, assim como não têm direito a proteção contra radiointerferências do Serviço de Radionavegação Aéronáutica;

III - de 5351,5 kHz a 5366,5 kHz, o limite não pode exceder a 25 watts (e.i.r.p.); e,

IV - de 10100 kHz a 10150 kHz, a potência média de operação na fração de tempo em que o sistema permanece ativo (duty cycle) não pode exceder 200 watts.

Art. 6º A potência média na saída do transmissor de uma estação terrestre de operação automática do Serviço de Radioamador deve estar limitada a 100 watts na fração de tempo em que o sistema permanece ativo (duty cycle), observados os limites específicos estabelecidos no art. 5º.

Parágrafo único. Considera-se estação automática aquela que prescinde da presença do titular, ou de conexão remota com ele, para seu funcionamento.

Art. 7º As estações do Serviço de Radioamador deverão observar as características básicas de emissão, as limitações específicas de potência, os planos de faixas com aplicações e demais especificações técnicas complementares estabelecidas por Atos da Superintendência responsável pela administração do uso do espectro de radiofrequências.

Parágrafo único. Os Atos referidos no caput serão submetidos ao procedimento de Consulta Pública antes de sua expedição.

Art. 8º O uso de modos de operação, larguras de faixa, aplicações e outras especificações não previstas no plano de faixas do serviço dependerão de autorização em caráter excepcional na Anatel, após apresentação devidamente fundamentada dos objetivos científicos ou experimentais e período de operação, cujo uso não poderá causar interferência nas aplicações originalmente previstas na respectiva subfaixa e faixas adjacentes.

Art. 9º A Anatel poderá solicitar aos interessados fundamentação específica para autorizar o uso das faixas de radiofrequências acima de 24 GHz.

CAPÍTULO III

DAS DISPOSIÇÕES FINAIS E TRANSITÓRIAS

Art. 10. As estações do Serviço de Radioamador deverão obedecer à legislação e à regulamentação específica sobre o uso de radiofrequências.

Art. 11. As estações do Serviço de Radioamador deverão ser licenciadas, nos termos da regulamentação específica sobre licenciamento de estações para telecomunicações.

Art. 12. As estações do Serviço de Radioamador deverão obedecer ao estabelecido na regulamentação específica sobre os limites de exposição a campos elétricos, magnéticos e eletromagnéticos.

Art. 13. Os equipamentos para radiocomunicações utilizados na exploração do Serviço de Radioamador, inclusive os sistemas radiantes, deverão cumprir os requisitos e observar o disposto na regulamentação específica sobre a certificação e homologação de produtos para telecomunicações.

Parágrafo único. Estão dispensados de atender aos requisitos mencionados no caput deste artigo os equipamentos produzidos de forma eventual ou artesanal e sem propósito comercial.

Art. 14. A Anatel poderá determinar alteração dos requisitos estabelecidos neste Regulamento, mesmo dos sistemas em operação, com a finalidade de otimizar o uso do espectro de radiofrequências.

Art. 15. Durante o período de vacância deste Regulamento, a Superintendência responsável pela administração do uso do espectro de radiofrequências deverá expedir os Atos de que trata o art. 7º, observado o disposto no parágrafo único daquele artigo.

SUPERINTENDÊNCIA DE FISCALIZAÇÃO
GERÊNCIA REGIONAL NO ESTADO
DE SÃO PAULO

ATO Nº 6.382, DE 22 DE AGOSTO DE 2018

Outorga autorização para uso de radiofrequência(s) à(o) SERVIMAR SERVICOS TECNICOS AMBIENTAIS LTDA., CNPJ nº 55.636.500/0001-06 associada à autorização para exploração do Serviço Limitado Privado.

DEBORA YAMADA
Gerente

ATO Nº 6.420, DE 23 DE AGOSTO DE 2018

Outorga autorização para uso de radiofrequência(s) à(o) AZUL LINHAS AERÉAS BRASILEIRAS S.A., CNPJ nº 09.296.295/0001-60 associada à autorização para exploração do Serviço Limitado Privado.

DEBORA YAMADA
Gerente

ATO Nº 6.522, DE 28 DE AGOSTO DE 2018

Outorga autorização para uso de radiofrequência(s) à(o) USINA BAZAN S/A, CNPJ nº 55.109.565/0001-01 associada à autorização para exploração do Serviço Limitado Privado.

SANDRO ALMEIDA RAMOS
Gerente

ATOS DE 27 DE AGOSTO DE 2018

Outorga autorização para uso de radiofrequência(s) associada à autorização para exploração do Serviço Limitado Privado à(o):

Nº 6.491 - PROTEGE S/A - PROTEÇÃO E TRANSPORTE DE VALORES, CNPJ nº 43.035.146/0029-86;

Nº 6.498 - GOCIL SERVICOS DE VIGILANCIA E SEGURANCA LTDA, CNPJ nº 50.844.182/0001-55;

Nº 6.502 - TIM CELULAR S.A., CNPJ nº 04.206.050/0001-80

SANDRO ALMEIDA RAMOS
Gerente

GERÊNCIA REGIONAL NOS ESTADOS DE GOIÁS,
MATO GROSSO, MATO GROSSO DO SUL
E TOCANTINS

UNIDADE OPERACIONAL
NO ESTADO DE MATO GROSSO DO SUL

ATO Nº 6.389, DE 22 DE AGOSTO DE 2018

Processo nº 53548.001094/2018-51. Expede autorização a TIAGO RAFAEL DE OLIVEIRA DIAS, CNPJ nº 01597142140, para explorar o Serviço Limitado Privado, por prazo indeterminado, sem caráter de exclusividade, em âmbito nacional e internacional e tendo como área de prestação de serviço todo o território nacional.

JOSÉ AFONSO COSMO JUNIOR
Gerente

ATO Nº 6.412, DE 23 DE AGOSTO DE 2018

Outorga autorização para uso de radiofrequências a CARLOS ALBERTO GODOY, CPF nº 475.730.821-34 associada à autorização para exploração do Serviço Limitado Privado.

JOSÉ AFONSO COSMO JUNIOR
Gerente

ANEXO B – Datasheet E32 433T30D

E32-TTL-1W Datasheet V1.0

Chengdu Ebyte Electronic Technology Co., Ltd.

Purchase the sample : <http://www.aliexpress.com/store/2077046>Website : www.cdebyte.com

4 . Instruction format

E32-TTL-1W

In sleep mode (mode 3 : M1=1, M0=1) , it supports below instructions on list.

(Only support 9600 and 8N1 format when setting)

No.	Instruction format	Illustration
1	C0 + working parameters	C0 + 5 bytes working parameters are sent in hexadecimal format. 6 bytes in total and must send in succession. (Save the parameters when power-down)
2	C1 C1 C1	Three C1 are sent in hexadecimal format. The module returns the saved parameters and must send in succession.
3	C2 + working parameters	C2 + 5 bytes working parameters are sent in hexadecimal format. 6 bytes in total and must send in succession. (Not save the parameters when power-down)
4	C3 C3 C3	Three C3 are sent in hexadecimal format. The module returns the version information and must send in succession.
5	C4 C4 C4	Three C4 are sent in hexadecimal format. The module will reset one time and must send in succession.

4.1 Default parameter

E32-TTL-1W

Default parameter values : C0 00 00 1A 17 44							
Model	Frequency	Address	Channel	Air data rate	Baud rate	Parity	Transmitting power
E32-TTL-1W	433MHz	0x0000	0x17	2.4kbps	9600	8N1	1W

4.2 Parameter setting instruction

E32-TTL-1W

The difference between C0 command and C2 command is that C0 command will write parameters into the internal flash memory and can be saved when power down, while C2 command cannot be saved when power down, because C2 command is temporarily mend instruction.

C2 is recommended for the occasion that need to change the operating parameters frequently,
Like C2 00 00 1A 17 44.

No.	Item	Description	Remark
0	HEAD	Fix 0xC0 or 0xC2, it means this frame data is control command	Must be 0xC0 or 0xC2 C0: Save the parameters when power-down C2: Not save the parameters when power-down
1	ADDH	High address byte of module (the default 00H)	00H-FFH
2	ADDL	Low address byte of module	00H-FFH

		(the default 00H)	
3	SPED	<p>Rate parameter , including UART baud rate and air data rate</p> <p>7 , 6 UART parity bit 00 : 8N1 (default) 01 : 8O1 10 : 8E1 11 : 8N1 (equal to 00)</p> <p>-----</p> <p>5 , 4 , 3 TTL UART baud rate (bps) 000 : 1200bps 001 : 2400bps 010 : 4800bps 011 : 9600bps (default) 100 : 19200bps 101 : 38400bps 110 : 57600bps 111 : 115200bps</p> <p>-----</p> <p>2 , 1 , 0 Air data rate (bps) 000 : 1Kbps (default) 001 : 2Kbps 010 : 5Kbps 011 : 8Kbps 100 : 10Kbps 101 : 15Kbps 110 : 20Kbps 111 : 25Kbps</p>	<ul style="list-style-type: none"> • UART mode can be different between communication parties <p>-----</p> <ul style="list-style-type: none"> • UART baud rate can be different between communication parties • The UART baud rate has nothing to do with wireless transmission parameters & won't affect the wireless transmit / receive features. <p>-----</p> <ul style="list-style-type: none"> • The lower the air data rate, the longer the transmitting distance, better anti-interference performance and longer transmitting time • The air data rate must keep the same for both communication parties.
4	CHAN	<p>7 , 6 , 5 : N/A</p> <p>-----</p> <p>4-0 : Communication channel, default 17H (433MHz)</p>	<ul style="list-style-type: none"> • 0(recommended) <p>-----</p> <p>00H-1FH</p>
5	OPTION	<p>7 , Fixed transmission (similar to MODBUS) 0 : Transparent transmission mode (default) 1 : Fixed transmission mode</p> <p>-----</p> <p>6 IO drive mode(the default 1) 1 : TXD and AUX push-pull outputs, RXD pull-up inputs</p>	<ul style="list-style-type: none"> • In fixed transmission mode, the first three bytes of each user's data frame can be used as high/low address and channel. The module changes its address and channel when transmit. And it will revert to original setting after complete the process. <p>-----</p> <p>This bit is used to the module internal pull-up resistor. It also increases the level's adaptability in case of open drain. But in some cases, it may need</p>

		<p>0 : TXD、AUX open-collector outputs, RXD open-collector inputs</p> <p>-----</p> <p>5 , 4 , 3 wireless wake-up time (for the receiver, it means the monitor interval time ,while for the transmitter it means continuously sending preamble code time.)</p> <p>000 : 250ms (default) 001 : 500ms 010 : 750ms 011 : 1000ms 100 : 1250ms 101 : 1500ms 110 : 1750ms 111 : 2000ms</p> <p>-----</p> <p>2 , FEC switch 0 : Turn off FEC 1 : Turn on FEC (Default)</p> <p>-----</p> <p>1, 0 transmission power (approximation) 00 : 30dBm (Default) 01 : 27dBm 10 : 24dBm 11 : 21dBm</p>	<p>external pull-up resistor.</p> <p>-----</p> <ul style="list-style-type: none"> • The transmit & receive module work in mode 0, whose delay time is invalid & can be arbitrary value. • The transmitter works in mode 1 can transmit the preamble code of the corresponding time continuously. • When the receiver works in mode 2, the time means the monitor interval time (wireless wake-up). Only the data from transmitter that works in mode 1 can be received. • The wake-up time set by transmitter cannot be less than the monitor interval time of receiver; otherwise, it may lead to data loss. In case of two-way communication, both parties should keep the wake-up time the same. • The longer the wake-up time, the lower the average receive current consumption. <p>-----</p> <ul style="list-style-type: none"> • After turn off FEC, the actual data transmission rate increases while anti-interference ability decreases. Also the transmission distance is relatively short. Both communication parties must keep on the same pages about turn-on or turn-off FEC. <p>-----</p> <ul style="list-style-type: none"> • The external power must make sure the ability of current output more than 200mA and ensure the power supply ripple within 100mV. Low power transmission is not recommended due to its low power supply efficiency.
--	--	--	--

For example: The meaning of No.3 "SPED" byte :

The binary bit of the byte	7	6	5	4	3	2	1	0
The specific value (user configures)	0	0	0	1	1	0	1	0
Meaning	UART parity bit 8N1		UART baud rate is 9600			Air data rate is 2.4K		
Corresponding hexadecimal	1				A			

4.3 Reading operating parameters

E32-TTL-1W

Instruction format	Description
C1+C1+C1	In sleep mode (M0=1 , M1=1) , User gives the module instruction (HEX format): C1 C1 C1, Module returns the present configuration parameters. For example, C2 00 00 1A 17 44.

4.4 Reading version number

E32-TTL-1W

Instruction format	Description
C3+C3+C3	In sleep mode (M0=1 , M1=1) , User gives the module instruction (HEX format): C3 C3 C3, Module returns its present version number, for example C3 32 xx yy. 32 here means the module model (E32 series); xx is the version number and yy refers to the other module features.

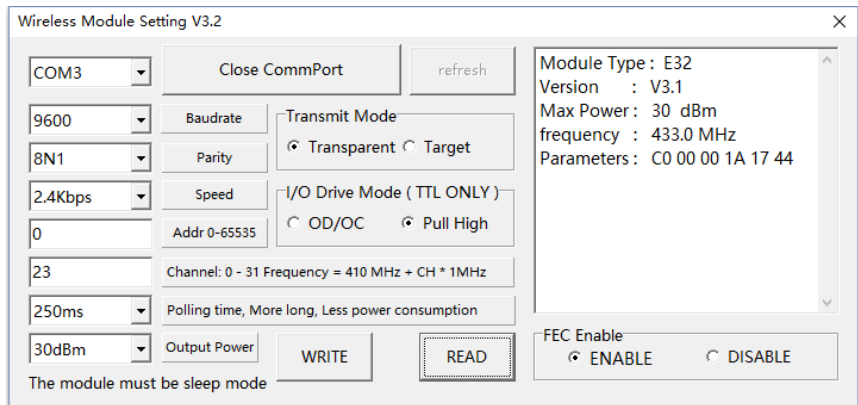
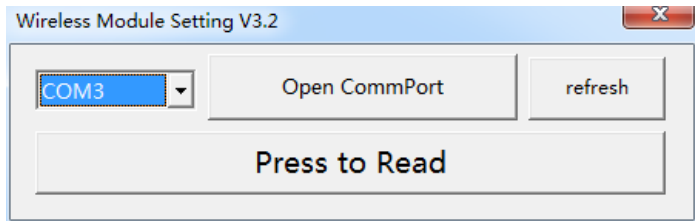
4.5 Reset instruction

E32-TTL-1W

Instruction format	Description
C4+C4+C4	In sleep mode (M0=1 , M1=1) , User gives the module instruction (HEX format): C4 C4 C4, the module resets for one time. During the reset process, the module will conduct self-check, AUX outputs low level. After reset completing, the AUX outputs high level, then the module starts to work regularly which the working mode can be switched or be given another instruction.

5 . Parameter setting **E32-TTL-1W**

Step	Operation	Description
1	Install Driver	Please install the USB adapter driver (CP2102).
2	Pull out the jumper	Pull the M0、 M1 jumper out, see figure 9 3.3V or 5V are available for jumper.
3	Connect to module	Connect the module with USB adapter. Connect to the USB interface of PC.
4	Open serial port	Operate the parameter setting software, choose corresponding serial number and press the "Open CommPort" button. Please choose other serial numbers until open successfully.
5	Interface	Press "Press to Read" button , the interface will be as figure 9 If failed, please check if the module is in mode 3, or the driver has been installed or not.
6	Input parameter	Please adjust the parameter as your request according to the corresponding setting, then click "Write" button, write the new parameter to the module
7	Complete the operation.	Please operate the "Fifth step" if you need to reconfigure, if the configuration is completed, Please click "close UART" and then take off the module.
8	Commands Configuration	Parameter configuration is also available for MCU (in mode 3).



ANEXO C – Datasheet GY450



ANTENA YAGI UHF

A Antena Yagi UHF, foi desenvolvida para fornecer um alto ganho. Seus elementos são soldados evitando assim possíveis maus contatos.

Está aterrada ao potencial D.C. para proteção contra raios.

Construção: Alumínio com tratamento anti-corrosão e pintura.

Fixação: Com Grampo Tipo "U", em cantoneira ou tubulão de até 2", ferragens galvanizada a fogo.

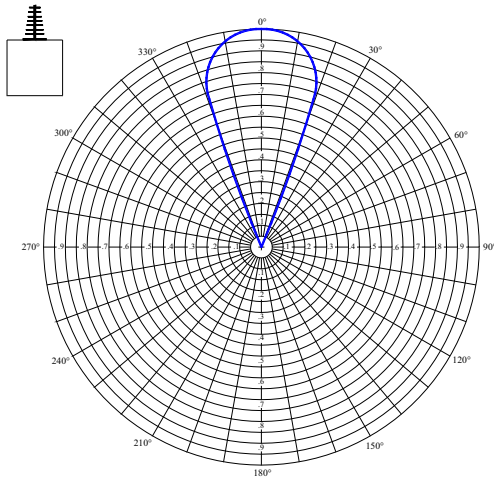
Alimentação: Dipolo Dobrado com Balun encapsulado com resina epoxy.



Imagem ilustrativa

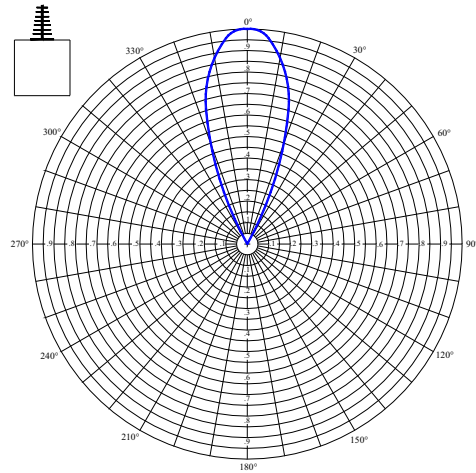
CARACTERÍSTICAS TÉCNICAS			
Descrição / Modelo		GY450	GYU
Faixa de Operação		430 a 470 MHz	14 a 69 (canal)
Largura de Faixa		40 MHz	12 MHz
Polarização		Horizontal / Vertical	
Numero de Elementos		7	16
Ganho	Vezes	6,10	27,22
	dBi	10,00	16,50
Ângulo à ± 3 dB	Horizontal	40°	32°
	Vertical	62°	36°
Relação Frente Costa		>22 dB	
Rej. de Polarização Cruzada		>20 dB	
VSWR		<1,5:1	< 1,2:1
Potência Máxima		250 Watts	
Impedância de Entrada		50 ou 75 Ohms	
Conector		N-fêmea= 50 Ohms e F-fêmea =75 Ohms	
Dimensões (mm)	Comprimento	1030	2700 (canal 14)
	Largura	350	270 (canal 14)
Peso (kg)		2,10	3,00
Vento de Resistência		130 Km/h	
Area de Exposição ao Vento		0,04 m ²	0,14 m ²

DIAGRAMA DE RADIAÇÃO HORIZONTAL
Escala E/Emax
GY450



Modelo		GY450
Ganho	Vezes	6,10
	dBi	10,00

DIAGRAMA DE RADIAÇÃO HORIZONTAL
Escala E/Emax
GYU



Modelo		GYU
Ganho	Vezes	27,23
	dBi	16,50

DIAGRAMA DE RADIAÇÃO VERTICAL
Escala E/Emax
Gy450

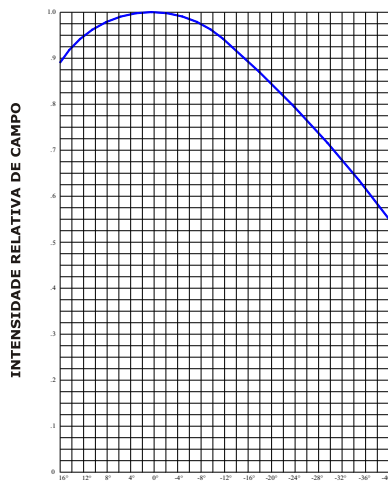
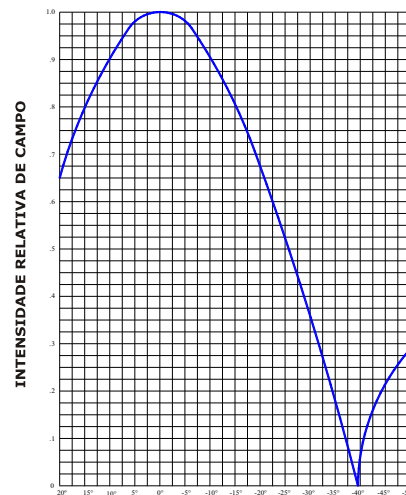


DIAGRAMA DE RADIAÇÃO VERTICAL
Escala E/Emax
GYU



ANEXO D – Datasheet ATmega328



Atmel 8-bit Microcontroller with 4/8/16/32KBytes In-System Programmable Flash

ATmega48A; ATmega48PA; ATmega88A; ATmega88PA;
ATmega168A; ATmega168PA; ATmega328; ATmega328P

SUMMARY

Features

- High Performance, Low Power Atmel®AVR® 8-Bit Microcontroller Family
- Advanced RISC Architecture
 - 131 Powerful Instructions – Most Single Clock Cycle Execution
 - 32 x 8 General Purpose Working Registers
 - Fully Static Operation
 - Up to 20 MIPS Throughput at 20MHz
 - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
 - 4/8/16/32KBytes of In-System Self-Programmable Flash program memory
 - 256/512/512/1KBytes EEPROM
 - 512/1K/1K/2KBytes Internal SRAM
 - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
 - Data retention: 20 years at 85°C/100 years at 25°C⁽¹⁾
 - Optional Boot Code Section with Independent Lock Bits
 - In-System Programming by On-chip Boot Program
 - True Read-While-Write Operation
 - Programming Lock for Software Security
- Atmel® QTouch® library support
 - Capacitive touch buttons, sliders and wheels
 - QTouch and QMatrix® acquisition
 - Up to 64 sense channels
- Peripheral Features
 - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
 - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
 - Real Time Counter with Separate Oscillator
 - Six PWM Channels
 - 8-channel 10-bit ADC in TQFP and QFN/MLF package
 - Temperature Measurement
 - 6-channel 10-bit ADC in PDIP Package
 - Temperature Measurement
 - Programmable Serial USART
 - Master/Slave SPI Serial Interface
 - Byte-oriented 2-wire Serial Interface (Philips I²C compatible)
 - Programmable Watchdog Timer with Separate On-chip Oscillator
 - On-chip Analog Comparator
 - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
 - Power-on Reset and Programmable Brown-out Detection
 - Internal Calibrated Oscillator
 - External and Internal Interrupt Sources
 - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
 - 23 Programmable I/O Lines
 - 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF
- Operating Voltage:
 - 1.8 - 5.5V
- Temperature Range:
 - -40°C to 85°C
- Speed Grade:
 - 0 - 4MHz@1.8 - 5.5V, 0 - 10MHz@2.7 - 5.5.V, 0 - 20MHz @ 4.5 - 5.5V
- Power Consumption at 1MHz, 1.8V, 25°C
 - Active Mode: 0.2mA
 - Power-down Mode: 0.1µA
 - Power-save Mode: 0.75µA (Including 32kHz RTC)

1. Pin Configurations

Figure 1-1. Pinout ATmega48A/PA/88A/PA/168A/PA/328/P

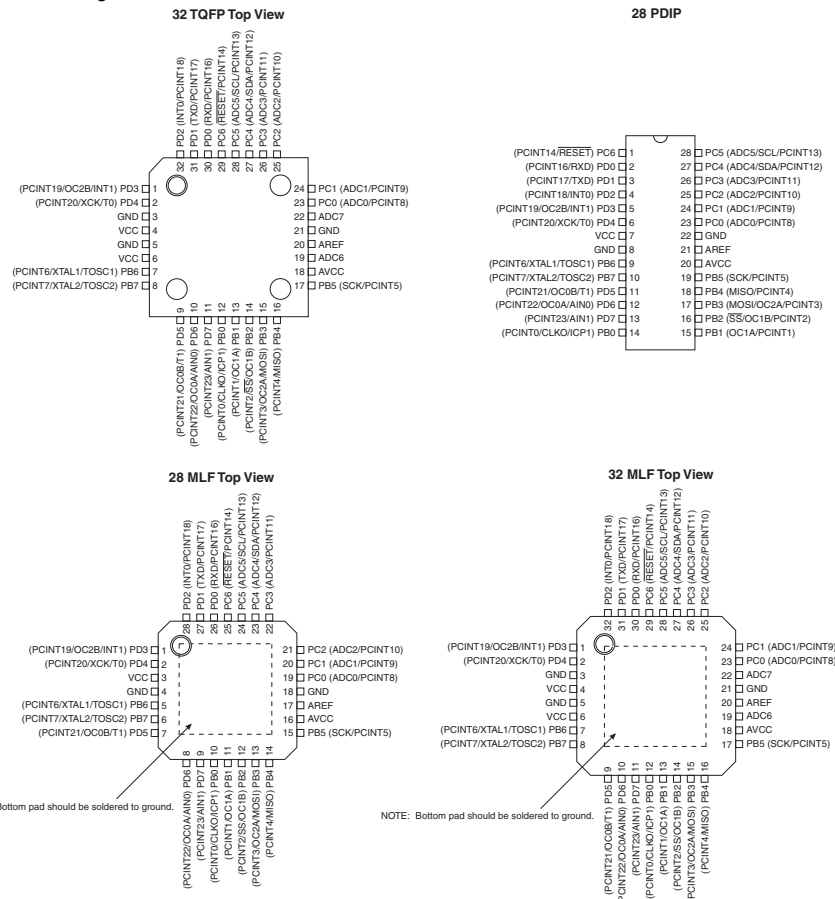


Table 1-1. 32UFPGA - Pinout ATmega48A/48PA/88A/88PA/168A/168PA

	1	2	3	4	5	6
A	PD2	PD1	PC6	PC4	PC2	PC1
B	PD3	PD4	PD0	PC5	PC3	PC0
C	GND	GND			ADC7	GND
D	VDD	VDD			AREF	ADC6
E	PB6	PD6	PB0	PB2	AVDD	PB5
F	PB7	PD5	PD7	PB1	PB3	PB4

1.1 Pin Descriptions

1.1.1 VCC

Digital supply voltage.

1.1.2 GND

Ground.

1.1.3 Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier.

If the Internal Calibrated RC Oscillator is used as chip clock source, PB7...6 is used as TOSC2...1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

The various special features of Port B are elaborated in "[Alternate Functions of Port B](#)" on page 83 and "[System Clock and Clock Options](#)" on page 26.

1.1.4 Port C (PC5:0)

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC5...0 output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

1.1.5 PC6/RESET

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C.

If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. The minimum pulse length is given in [Table 29-12 on page 310](#). Shorter pulses are not guaranteed to generate a Reset.

The various special features of Port C are elaborated in "[Alternate Functions of Port C](#)" on page 86.


1.1.6 Port D (PD7:0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

The various special features of Port D are elaborated in "[Alternate Functions of Port D](#)" on page 89.

1.1.7 AV_{CC}

AV_{CC} is the supply voltage pin for the A/D Converter, PC3:0, and ADC7:6. It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter. Note that PC6...4 use digital supply voltage, V_{CC}.



1.1.8 AREF

AREF is the analog reference pin for the A/D Converter.

1.1.9 ADC7:6 (TQFP and QFN/MLF Package Only)

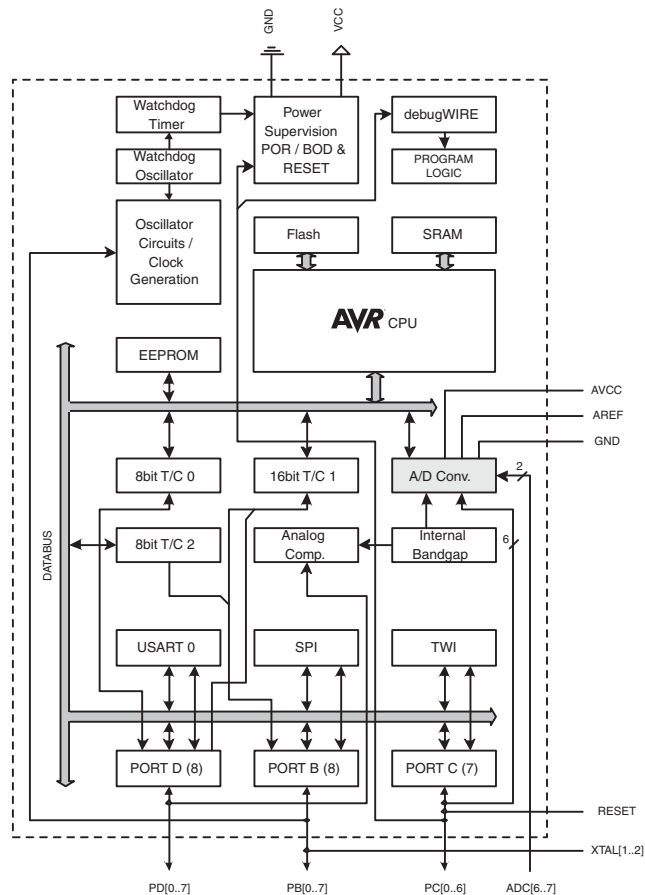
In the TQFP and QFN/MLF package, ADC7:6 serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

2. Overview

The ATmega48A/PA/88A/PA/168A/PA/328/P is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega48A/PA/88A/PA/168A/PA/328/P achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram

Figure 2-1. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.



The ATmega48A/PA/88A/PA/168A/PA/328/P provides the following features: 4K/8Kbytes of In-System Programmable Flash with Read-While-Write capabilities, 256/512/512/1Kbytes EEPROM, 512/1K/1K/2Kbytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte-oriented 2-wire Serial Interface, an SPI serial port, a 6-channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages), a programmable Watchdog Timer with internal Oscillator, and five software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, USART, 2-wire Serial Interface, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption.

Atmel® offers the QTouch® library for embedding capacitive touch buttons, sliders and wheels functionality into AVR® microcontrollers. The patented charge-transfer signal acquisition offers robust sensing and includes fully debounced reporting of touch keys and includes Adjacent Key Suppression® (AKS™) technology for unambiguous detection of key events. The easy-to-use QTouch Suite toolchain allows you to explore, develop and debug your own touch applications.

The device is manufactured using Atmel's high density non-volatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional non-volatile memory programmer, or by an On-chip Boot program running on the AVR core. The Boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega48A/PA/88A/PA/168A/PA/328/P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega48A/PA/88A/PA/168A/PA/328/P AVR is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, Program Debugger/Simulators, In-Circuit Emulators, and Evaluation kits.


2.2 Comparison Between Processors

The ATmega48A/PA/88A/PA/168A/PA/328/P differ only in memory sizes, boot loader support, and interrupt vector sizes. [Table 2-1](#) summarizes the different memory and interrupt vector sizes for the devices.

Table 2-1. Memory Size Summary

Device	Flash	EEPROM	RAM	Interrupt Vector Size
ATmega48A	4KBytes	256Bytes	512Bytes	1 instruction word/vector
ATmega48PA	4KBytes	256Bytes	512Bytes	1 instruction word/vector
ATmega88A	8KBytes	512Bytes	1KBytes	1 instruction word/vector
ATmega88PA	8KBytes	512Bytes	1KBytes	1 instruction word/vector
ATmega168A	16KBytes	512Bytes	1KBytes	2 instruction words/vector
ATmega168PA	16KBytes	512Bytes	1KBytes	2 instruction words/vector
ATmega328	32KBytes	1KBytes	2KBytes	2 instruction words/vector
ATmega328P	32KBytes	1KBytes	2KBytes	2 instruction words/vector

ATmega48A/PA/88A/PA/168A/PA/328/P support a real Read-While-Write Self-Programming mechanism. There is a separate Boot Loader Section, and the SPM instruction can only execute from there. In ATmega 48A/48PA there



is no Read-While-Write support and no separate Boot Loader Section. The SPM instruction can execute from the entire Flash

3. Resources

A comprehensive set of development tools, application notes and datasheets are available for download on <http://www.atmel.com/avr>.

4. Data Retention

Reliability Qualification results show that the projected data retention failure rate is much less than 1 PPM over 20 years at 85°C or 100 years at 25°C.

5. About Code Examples

This documentation contains simple code examples that briefly show how to use various parts of the device. These code examples assume that the part specific header file is included before compilation. Be aware that not all C compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C compiler documentation for more details.

For I/O Registers located in extended I/O map, “IN”, “OUT”, “SBIS”, “SBIC”, “CBI”, and “SBI” instructions must be replaced with instructions that allow access to extended I/O. Typically “LDS” and “STS” combined with “SBRS”, “SBRC”, “SBR”, and “CBR”.

6. Capacitive Touch Sensing

The Atmel® QTouch® Library provides a simple to use solution to realize touch sensitive interfaces on most Atmel AVR® microcontrollers. The QTouch Library includes support for the Atmel QTouch and Atmel QMatrix® acquisition methods.

Touch sensing can be added to any application by linking the appropriate Atmel QTouch Library for the AVR Microcontroller. This is done by using a simple set of APIs to define the touch channels and sensors, and then calling the touch sensing API's to retrieve the channel information and determine the touch sensor states.

The QTouch Library is FREE and downloadable from the Atmel website at the following location: www.atmel.com/qtouchlibrary. For implementation details and other information, refer to the [Atmel QTouch Library User Guide](#) - also available for download from Atmel website.

8.7 ATmega328

Speed (MHz)	Power Supply (V)	Ordering Code ⁽²⁾	Package ⁽¹⁾	Operational Range
20 ⁽³⁾	1.8 - 5.5	ATmega328-AU ATmega328-AUR ⁽⁵⁾ ATmega328-MMH ⁽⁴⁾ ATmega328-MMHR ⁽⁴⁾⁽⁵⁾ ATmega328-MU ATmega328-MUR ⁽⁵⁾ ATmega328-PU	32A 32A 28M1 28M1 32M1-A 32M1-A 28P3	Industrial (-40°C to 85°C)

- Note:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
 2. Pb-free packaging complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
 3. See [Figure 29-1 on page 308](#).
 4. NiPdAu Lead Finish.
 5. Tape & Reel

Package Type	
32A	32-lead, Thin (1.0mm) Plastic Quad Flat Package (TQFP)
28M1	28-pad, 4 x 4 x 1.0 body, Lead Pitch 0.45mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)
28P3	28-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)
32M1-A	32-pad, 5 x 5 x 1.0 body, Lead Pitch 0.50mm Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)