9º Workshop em Engenharia e Tecnologia Espaciais
15 e 16 de Agosto de 2018

Simulação da operação e comunicação de uma carga útil baseada na Sonda de Langmuir com o OBC do NanosatC-Br2

ALMEIDA, D.P.[1], ABELHA, M.[2], ARPINO, C.[3], COELHO, L.[4], CORRÊA, M.[5], SARAIVA, J.P.[6], MATTIELLO-FRANCISCO, F[1].
[1]Instituto Nacional de Pesquisas Espaciais, São José dos Campos, SP, Brasil
[2]Universidade Federal de Minas Gerais, Belo Horizonte, MG, Brasil
[3]Universidade Federal do Rio Grande do Sul, Porto Alegre, RS, Brasil
[4]Instituto Federal de Santa Catarina, São José, SC, Brasil
[5]Universidade Federal de Itajubá, Itajubá, MG, Brasil
[6]Universidade Federal de São Paulo, São José dos Campos, SP, Brasil

danilo.pallamin@inpe.br

Resumo. *Este artigo apresenta o desenvolvimento de uma simulação de uma das cargas úteis do NanosatC-BR2 de modo a implementar a operação da carga útil pelo computador de bordo e o protocolo de comunicação utilizado no satélite com as cargas úteis (I2C), como mini-estágio do 14° Curso de Inverno de Introdução às Tecnologias Espaciais, oferecido pelo INPE em Julho de 2018, com objetivo de síntese e aplicação de conhecimentos adquiridos ao longo do curso. Para tanto, foi utilizado o modelo de engenharia do Computador de Bordo do satélite (iOBC) com uma placa de desenvolvimento Arduino Uno para simular a carga útil. Os resultados do trabalho são as funções concebidas em conjunto pela equipe para a realização da operação da carga útil considerando as capacidades e limitações do protocolo I2C.*

Palavras-chave: I2C; Onboard Software; CubeSat; NanosatC-Br2; Sonda de Langmuir

1. Introdução

Este artigo demonstra o trabalho e os resultados obtidos através do estudo e implementação da simulação de uma carga útil baseada na Sonda de Langmuir, carga útil do NanosatC-BR2 que está sendo desenvolvida pelo INPE, como parte do mini-estágio oferecido como conclusão do Curso de Inverno de Introdução a Tecnologias Espaciais realizado em Julho de 2018.

O objetivo deste trabalho foi de analisar a carga útil em etapa final de desenvolvimento através do último relatório da equipe de desenvolvimento, e criar um simulador em um microcontrolador de um sistema similar com operação análoga para auxiliar na concepção da operação da carga útil real.

Objetivos secundários, como parte do Curso de Inverno e dirigidos aos alunos, foram estudar:

- Tecnologias CubeSat-related e os sistemas utilizados
- Desenvolvimento de Software de Bordo de sistemas embarcados
- Operação de cargas úteis
- Protocolo I2C de comunicação digital

Os CubeSats surgiram com a proposta de auxiliar no ensino e aprendizagem em unidades curriculares relacionadas ao projeto, construção, teste e operação de satélites no espaço. A proposta inicial surgiu como uma idéia de padronização de conceito e tinha como pressuposto que alunos de pós-graduação, principalmente, passassem por todas as etapas do desenvolvimento de um satélite. Com o sucesso do padrão, CubeSats foram adotados por diversas instituições no mundo, incluindo empresas privadas, para finalidades não apenas de ensino, bem como comerciais e de validação tecnológica, entre outras [1].

O NanosatC-Br2 é um nanosatélite concebido através da parceria entre o Centro Regional Sul (CRS) do Instituto Nacional de Pesquisas Espaciais (INPE) e a Universidade Federal de Santa Maria - UFSM pelo convênio MCTIC/INPE-UFSM, que segue o padrão CubeSat.

O satélite é dividido em dois módulos, que ocupam 1U cada: o *módulo de serviço*, constituído pelos subsistemas ligados à operação do satélite, e o *módulo de carga útil*, composto pelas cargas úteis físicas da missão. Na Figura 1, Conceição [2] mostra o Modelo de Engenharia do satélite:

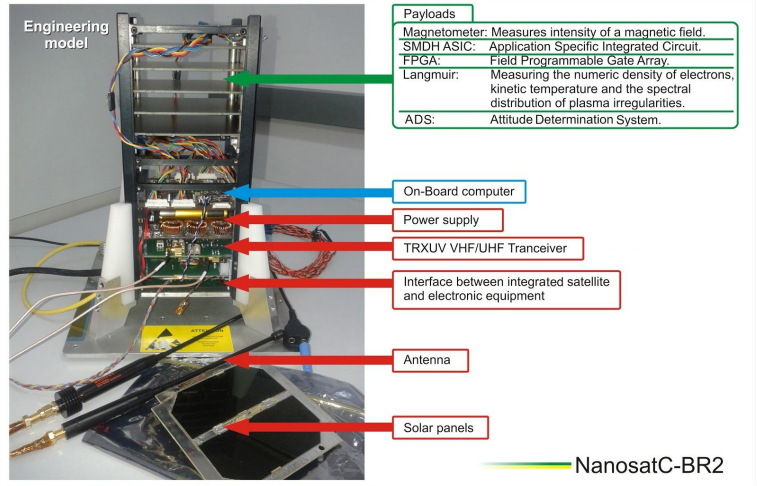

Figura 1: Modelo de Engenharia do NanosatC-BR2

2. Metodologia

A etapa final de desenvolvimento do satélite, enquadrada na Fase C do ciclo de vida de um projeto na Engenharia de Sistemas [3], engloba o desenvolvimento do software do computador de bordo do satélite, e é onde está situado o NanosatC-BR2 e onde se enquadra este trabalho.

Os alunos dividiram-se em dois grupos, um como desenvolvedores do computador de bordo (OBC) e outro como desenvolvedores da carga útil (SLPCI). Após cada equipe estudar seu respectivo subsistema, foi feita uma reunião em que definiram-se as funções para a comunicação e operação da carga útil, que deveriam ser implementadas para a operação seguir o fluxograma do algoritmo proposto pelos desenvolvedores da carga útil real.

Para criar a SLPCI, foi utilizado uma plataforma de desenvolvimento Arduino Uno, que utiliza um microcontrolador Atmel ATMega328. Para o computador de bordo, foi utilizado o modelo de engenharia do ISIS Onboard Computer (iOBC), parte do NanosatC-BR2, que utiliza um microprocessador ARM9 de 32 bits AT91SAM9G20.

Foram estudados e utilizados os ambientes de desenvolvimento de cada plataforma. Para o Arduino, foi utilizada a própria IDE do Arduino, que já inclui a própria linguagem, baseada em C++, seu próprio compilador e gravador integrados. Para o iOBC, foi utilizado o Eclipse provido pela Customer Package da fabricante, incluindo compilador e gravador JLINK usando um SAM-ICE da Atmel. O software é escrito em C usando uma HAL (Hardware Abstraction Layer) própria da fabricante específica para a iOBC, inclusa no Customer Package.

O protocolo de comunicação implementado foi o I2C, que é o utilizado no NanosatC-Br2. Para isto, foi-se estudado o protocolo e sua implementação nas plataformas.

3. Descrição do Sistema

3.1. Carga útil Sonda de Langmuir

A Sonda de Langmuir (SLP) é um instrumento utilizado para medir a densidade numérica de elétrons, a temperatura cinética, e a distribuição espectral das irregularidades de plasma. A carga útil que está sendo desenvolvida no INPE para ser embarcada no NanosatC-BR2, opera de maneira em que os dados lidos pelos sensores são armazenados em um *buffer* interno de RAM (preenchendo 1 pacote de 100 B por segundo) até que este é preenchido totalmente, que é então transferido para o computador de bordo do satélite [4].

A simulação da sonda (SLPCI) criada neste projeto compreende a criação de um software operando no Arduino que recrie similarmente a operação da sonda real, baseada no último relatório da equipe desenvolvedora da sonda, que inclui a criação de um *buffer* com dados fictícios, dentro dos limites de memória da plataforma Arduino, com as funções e comandos necessários para a geração e transmissão do *buffer* para o computador de bordo.

3.2. Comunicação I²C

A comunicação entre a SBCI e o OBC foi feita através do barramento I²C (Inter-Integrated Circuit), onde a transmissão dos dados é feita bidirecionalmente pelo canal Serial Data (SDA), utilizando a sincronização feita pelo canal Serial Clock (SCL). Na

comunicação I2C a velocidade de transmissão (baud) pode ser variável dependendo da aplicação. Neste caso, optou-se pelo padrão de 100 kbps que é aceito tanto pelo Arduino quanto pelo iOBC.

I2C é um protocolo que segue o sistema Mestre-Escravo, em que permite-se apenas um Mestre, porém diversos escravos. Cada escravo possui um endereço, em que o sistema de endereçamento pode ser de 8 bits (permitindo 255 escravos) ou mais. O mestre é o único que inicia as transações no barramento, e cada transação tem número finito de pacotes de palavras que são enviados. Para que o Escravo envie pacotes no barramento, o mesmo deve ser solicitado previamente pelo Mestre, que assim governa as transações que são sempre entre Mestre e Escravo, não podendo ter informação sendo enviada de Escravo para Escravo.

O funcionamento do I2C é através de interrupção, então quando o driver I2C de cada elemento detecta dados no buffer do canal, o processador interrompe o que estava executando para executar as funções relacionadas com leitura e envio do protocolo.

A implementação do I2C no iOBC opera basicamente com as funções:

- **i2c_write(address,data,size):** Em que é enviado o pacote de dados **data** de tamanho **size** bytes para o escravo no endereço **address**.
- **i2c_read(address,data,size):** Em que solicita um pacote de dados **data** de tamanho **size** bytes do escravo no endereço **address**.

No Arduino, a implementação de I2C no modo escravo possui duas funções de interrupção:

- **receiveEvent():** para quando o comando recebido é de quando o Mestre está enviando pacotes de dados.
- **requestEvent():** para quando o comando recebido é de quando o Mestre solicita o envio de pacotes de dados.

No caso específico do Arduino, e portanto a limitação do sistema, o limite do *buffer* de transmissão de dados por transação é de 32 Bytes.

A tensão de operação do I2C pode ser de 5V (como no Arduino) ou 3.3 (como no iOBC). Para a compatibilização, deve haver, portanto, um sistema de conversão de nível de tensão.

3.3. Hardware

O trabalho realizado utilizou um microprocessador ARM9 32-bit (AT91SAM9G20) presente no iOBC, um microcontrolador Atmel ATmega328 da placa Arduino UNO, materiais de laboratório (osciloscópio, fonte de bancada e multímetro), o EGSE (electrical ground support equipment) da ISIS para o modelo de engenharia do Nanosat, materiais de instrumentação (protoboard e jumpers) e um programador in-circuit debugger SAM-ICE.

A implementação da comunicação pelo protocolo I2C necessitou de um circuito conversor de tensão de 5V do Arduino para 3.3V no iOBC. Na Figura 2 é possível verificar a demonstração da montagem do circuito regulador em conjunto com o Arduino UNO utilizado na simulação da operação de telecomando.

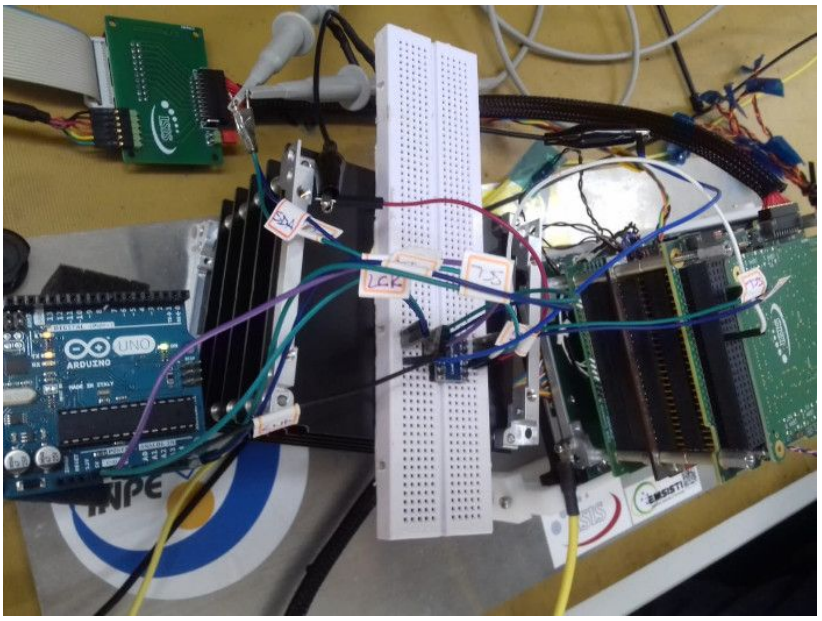

Figura 2. Setup do conversor de tensão com o iOBC e o Arduino

3.4. Software

O desenvolvimento dos softwares dos subsistemas foi feito de modo a seguir o diagrama de sequência ilustrado na Figura 3, para a operação da carga útil pelo computador de bordo:

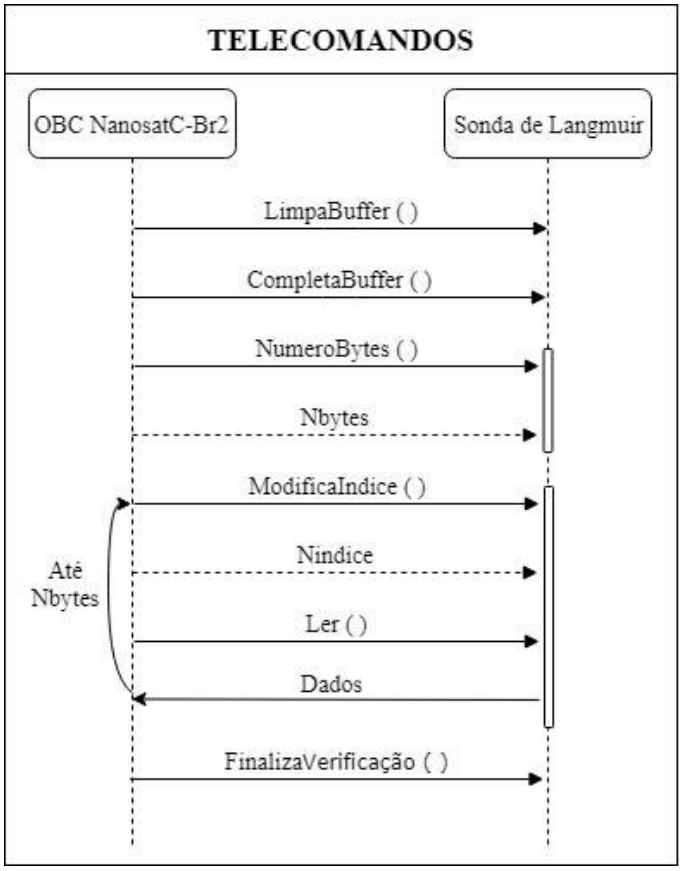

Figura 3. Diagrama de sequência da operação desejada

Inicialmente, o computador de bordo solicita o esvaziamento do buffer. Em seguida, envia o comando para que a carga útil faça a aquisição dos dados e preencha o buffer com as leituras simuladas. Após isso, envia a quantidade de bytes a serem transmitido por pacote de transação (podendo ser modificado ao longo da operação) e aí inicia a coleta do buffer. A coleta é feita através da solicitação de Nbytes à partir do índice I do buffer, portanto o computador de bordo repete esta solicitação, atualizando o índice desejado a cada iteração, até que o buffer esteja completo. A cada solicitação, a sonda responde com os dados de acordo com a quantidade solicitada.

3.4.1. Software - Carga útil

A tarefa principal do software da carga útil é preencher um buffer de tamanho pré estabelecido (neste caso, 1KB para respeitar limites do Arduino), e disponibilizar as funções para que o computador de bordo colete este buffer.

A função RequestEvent, representada pelo diagrama da Figura 4, é responsável pelo retorno de informações do buffer, estando dividida em dois casos. O primeiro caso ocorre quando a última escrita do mestre (OBC) é definição do índice (0xF3), o retorno da função é a parte do buffer que inicia no índice e termina na posição equivalente a quantidade de bytes ambos determinados pelo mestre (0xF2). No segundo, quando a última operação realizada seja verificar disponibilidade do buffer (0xF4), se não estiver sendo carregado ou limpo, está disponível (process_end igual à 1) como podemos ver na Figura 3 e retorna 1, caso contrário retorna 0.

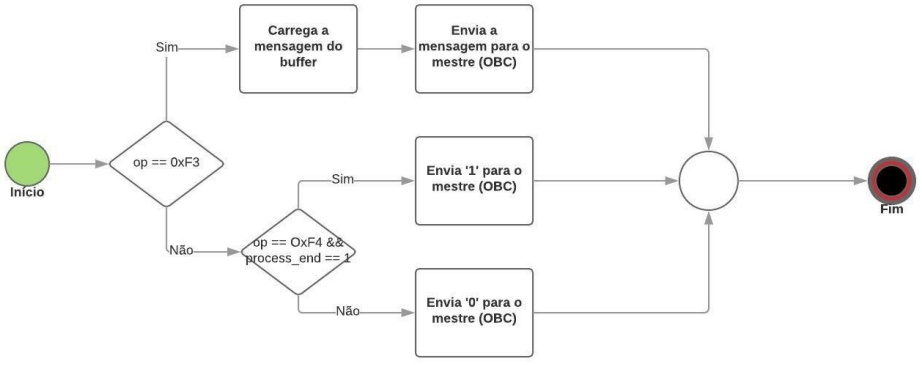

Figura 4. Diagrama lógico da RequestEvent

A função ReceiveEvent realiza as operações definidas pelo OBC, que são: Limpar o buffer (0xF0), carregar o buffer (0xF1), ler a quantidade de bytes (0xF2) e ler o índice (0xF3). Nas operações de limpeza e carregamento do buffer, o escravo apenas identifica a solicitação, chama as funções específicas para realizá-las e sinaliza o buffer como indisponível (process_end igual à zero) quando elas estão sendo executadas e disponível após finalizá-las (process_ end igual à um) , como podemos ver na Figura 4 .

A operação de leitura do índice e quantidade de bytes é feita dentro da função ReceiveEvent, quando selecionada essas opções o valor de uma flag de controle (select_function) é alterado de acordo com a quantidade de bytes que será necessário para realizar as leituras, no caso do índice 3 caracteres (bytes) e da quantidade de bytes 2

caracteres (bytes), e então o segundo caso em que é utilizado um laço para realizar a leitura desses caracteres correspondente aos números.

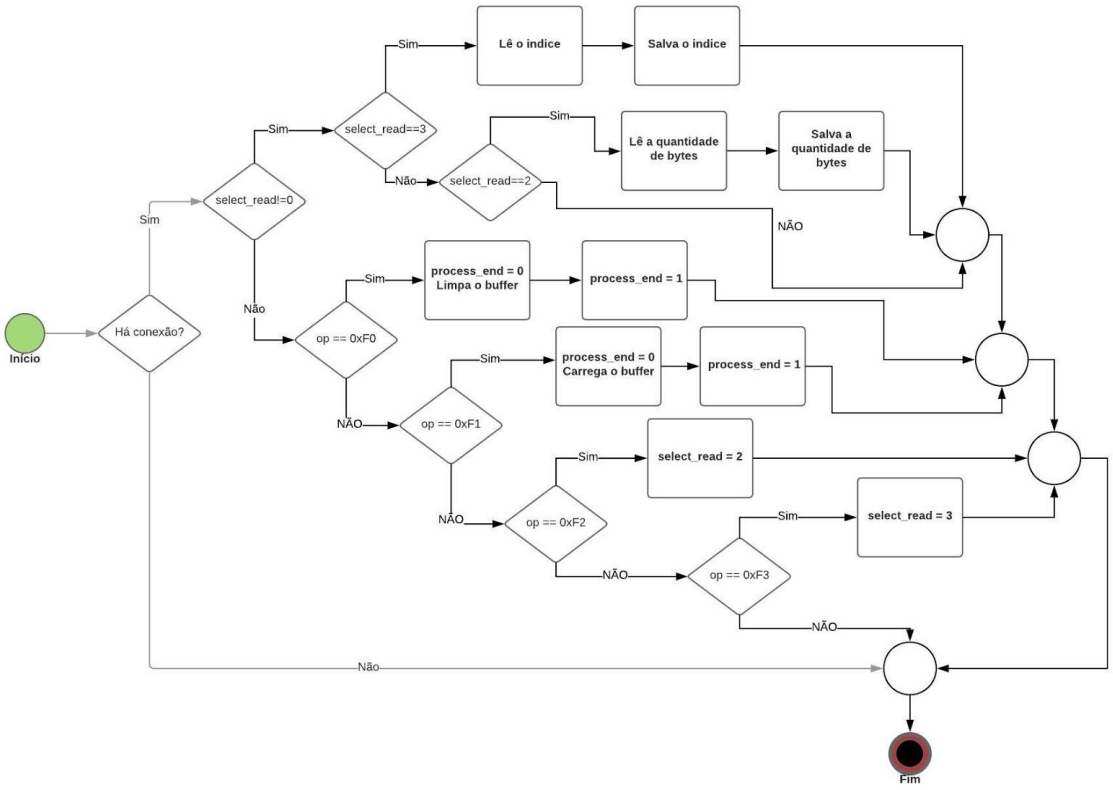

Figura 5. Diagrama lógico do ReceiveEvent

3.4.2. Software - On-board Computer

O software do computador de bordo, programado em linguagem C, utiliza da HAL da fabricante do iOBC para utilizar os periféricos e funções internas do iOBC, como o próprio driver de I2C.

Em sua função principal, é inicialmente solicitado à carga útil que limpe seu buffer e inicie a coleta de dados, e em seguida é criado um loop de chamadas de funções de leituras do buffer da carga útil sequencialmente até que o buffer inteiro seja coletado. Para simplificar o preenchimento do buffer de 1000 Bytes, foram utilizadas transações de 10 e 20 Bytes como exemplos.

Após a coleta do buffer de dados, o buffer foi impresso via terminal serial (usado o software Putty) para a visualização e comparação do buffer recebido.

4. Resultados e Discussão

A operação da carga útil simulada foi bem sucedida, resultando na transferência correta do buffer de dados gerado no Arduino para o iOBC. Na Figura 6 pode-se observar os dois terminais Serial, do Putty (iOBC) à esquerda, e do Arduino à direita, comparando os dois buffers e as mensagens enviadas e recebidas.

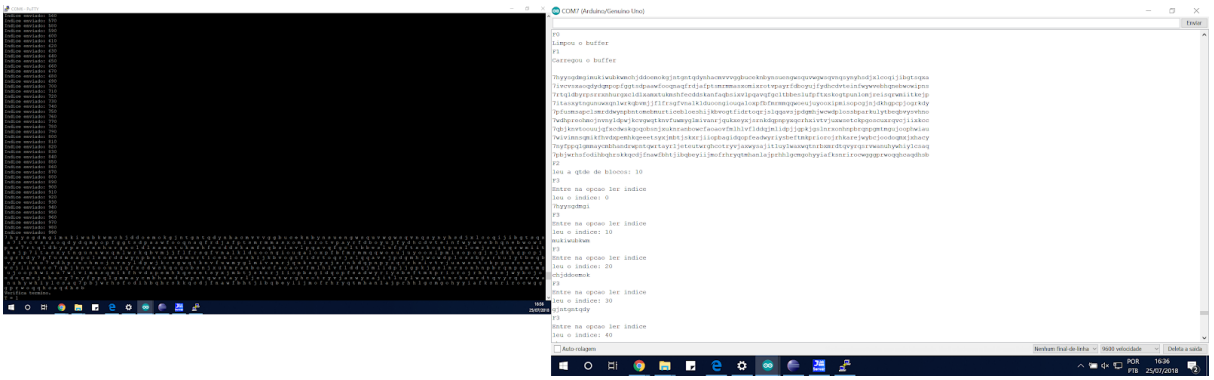

Figura 6. Terminais Serial dos subsistemas

As equipes foram capazes de sintetizar a descrição da operação da carga útil real e criar uma carga útil similar, criando as funções e operação de acordo com o esperado. Também foi evidenciado o bom entendimento e implementação do protocolo I2C, amplamente utilizado no universo de sistemas embarcados e nanossatélites.

5. Conclusão

A equipe completou o objetivo de simular a operação da carga útil estudada, e com isso pôde se aprofundar nas tecnologias da área de sistemas embarcados utilizadas em satélites, como os subsistemas existentes, o protocolo de comunicação, e a carga útil em si, bem como nas aplicações de CubeSats, sintetizando com a experiência aplicada parte dos conhecimentos adquiridos durante o Curso de Inverno de Introdução às Tecnologias Espaciais, que era o objetivo do mini-estágio.

Agradecimentos: *Agradecemos à CAPES pelo fomento da pós graduação, ao Curso de Inverno ETE/INPE e à equipe do NanosatC-BR2 por providenciar a oportunidade .*

Referências

[1] HOWELL, Elizabeth. **CubeSats: Tiny Payloads, Huge Benefits for Space Research.** Junho, 2018. https://www.space.com/34324-cubesats.html (Acessado em Agosto/2018)

[2] CONCEIÇÃO, Carlos A. P. **On the use of NanosatC-Br Test System for Payload Operational Requirements Verification**. II Latin American IAA CubeSat Workshop, Florianópolis, Brazil, 2016.

[3] NASA Systems Engineering Handbook NASA SP-2016-6105 Rev2

[4] ALMEIDA, Danilo. **Modelagem da Interoperabilidade entre Computador de Bordo e Cargas Úteis do NanosatC-Br2 com apoio da Ferramenta UPPAAL e Análise & Projeto da Daughterboard do Computador de Bordo**. Trabalho de Conclusão de Curso apresentado da Escola de Engenharia de São Carlos, Universidade de São Paulo, 2016.